AUTHENTICATION AND KEY ESTABLISHMENT IN WIRELESS NETWORKS

ZHIGUO WAN

NATIONAL UNIVERSITY OF SINGAPORE

2006

AUTHENTICATION AND KEY ESTABLISHMENT IN WIRELESS NETWORKS

ZHIGUO WAN
(B.S., Tsinghua University)

A THESIS SUBMITTED

FOR THE DEGREE OFDOCTOR OF PHILOSOPHY

SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF SINGAPORE

2006

Acknowledgments

It is a long journey from the time I started my research on wireless network security until I finally finished this dissertation. This long process is full of painful frustration, hard work, and cheerful excitement. As all these things are going to reach an end, it is time for me to express my gratitude to those people who have helped and contributed to my research work all these years.

First of all, I would like to thank my supervisor Prof. Robert H. Deng. It is Prof. Deng that guide me into the research field of wireless network security. He has been a wonderful advisor, giving me good suggestions and guidance with patience. I am really grateful for those hours he spent on discussing research topics and amending papers with me, which is crucial to me. His breadth of knowledge and enthusiasm for research always inspires me. These years of studying under his supervision is highly valuable in my life.

From the bottom of my heart, I want to express my gratitude to my co-supervisor Dr. Feng Bao. Dr. Bao is a great supervisor on advising students in research. My first published paper was completed under his supervision, which has been my precious experience on research. I benefited a lot from discussion with Dr. Feng Bao, and his insight into research in security has inspired me.

I would especially like to thank my co-supervisor Prof. Akkihebbal L. Ananda. Prof. Ananda has been an admirable and wonderful advisor, giving me valuable suggestions

for my papers. From the start of my candidature, Prof. Akkihebbal Ananda has helped me with my qualification exam, thesis proposal, final thesis submission, and job hunting.

A lot of people in Infocomm Security Department of I2R have been helpful to me and enriched my life here: Yang Yanjiang, Zhu Bo, Ren Kui, Wang Shuhong, Li Shiqun, Qi Fang, Chen Xiangguo, Guo Lifeng, Liu Yang, and Shane Balfe, who visited I2R for half a year. I am really grateful to them for their help and valuable discussion on various research topics.

I am deeply indebted to National University of Singapore, which provides me scholarship for all these years and such a wonderful research environment. My study in NUS would become one part of my most precious memory, and I would never forget the kindness offered by NUS.

Finally, I would like to thank my family, my parents and my sister, for their love and support. They are always supportive and encourage me when I am depressed with frustration. I am most grateful for everything they have done for me.

Table of Contents

List of Tables

List of Figures

Abbreviation List

2G	Second Generation
3G	Third Generation
AMP	Authentication and key agreement via Memorable Passwords
AODV	Ad hoc On-demand Distance Vector Routing
BD	Burmester-Desmedt Protocol
DoS	Denial of Service
EAP	Extensible Authentication Protocol
EKE	Encrypted Key Exchange
GDH	Group Diffie-Hellman
GSM	Global System for Mobile Communication
ICV	Integrity Check Value
IKE	Internet Key Exchange
JFK	Just Fast Keying
LAN	Local Area Network
LKH	Logical Key Hierarchy
MAC	Medium Access Control
OFT	One-way Function Tree
PAK	Password-Authenticated Key Exchange
PAKE	Password-Authenticated Key Exchange
2PAKE	2-party PAKE
nPAKE	n-party PAKE
PKC	Public Key Cryptosystem
SIAP	Secure Internet Access Protocol
SLAP	Secure Link Access Protocol
SRP	Secure Remote Password Protocol
STR	Steer et al. Protocol
SPEKE	Simple Password Exponential Key Exchange

TGDH	Tree-based Group Diffie-Hellman
TTP	Trusted Third Party
WEP	Wired Equivalent Privacy
WLAN	Wireless Local Area Network
WMAN	Wireless Metropolitan Area Network
WMN	Wireless Mesh Network
WPAN	Wireless Personal Area Network
WWAN	Wireless Wide Area Network

Summary

As the trend toward a ubiquitous computing world is gaining momentum, concern about security in wireless networks has become the major obstacle of their extensive applications. Due to their unique characteristics, wireless networks are more vulnerable against different attacks than their wired counterpart.

Different security protocols have been proposed and investigated to counter against security attacks in wireless networks. Essentially, these protocols can be classified into two groups: two-party key exchange protocols, and multi-party key management protocols (a.k.a. group key management protocols). In this thesis, we investigated both two-party and multi-party security protocols for wireless networks.

We first studied two-party authentication and key exchange protocols for access control in wireless networks in public places. Our analysis shows that previous access control protocols have serious security flaws which make them vulnerable to attacks. Then we proposed a password-based protocol and a PKC-based protocol under the two-layer access control architecture, respectively. Both of our protocols avoid weaknesses of previous proposals and provide mutual authentication, perfect forward secrecy, access control on wireless networks. Moreover, they also provide DoS resistance and identity anonymity for clients. We presented detailed security and performance analysis for our protocols, which showed that both our protocols are secure and efficient for access control in wireless networks.

We then studied multi-party key management protocols for wireless networks. We proposed a highly efficient group key agreement scheme based on a novel key tree construction approach for wireless ad hoc networks. The key tree is constructed taking into consideration of the multicast tree which represents the underlying network topology. Our scheme greatly reduces communication and computation cost for group key agreement and has high flexibility in handling dynamic group memberships. We implemented our scheme on ns-2 and evaluated its performance in terms of total delay, communication cost and message loss. Our simulation results show that the scheme enjoys great advantages over existing schemes proposed in the literature.

An efficient password-only group key agreement protocol is also proposed for wireless networks. In this scheme, each user shares a human-memorable password with a trusted server, and a group of users from a multi-hop wireless network intend to agree on a group key with the server's assistance. Our password-based group key agreement protocol achieves communication and computation efficiency, as a group key tree well-suited for multi-hop wireless networks is specially designed for group key agreement. With our protocol, a group of users can agree on a group key within only 3 flows, and each user needs only $5 + O(\log n)$ exponentiations.

In this thesis, the two proposed access control schemes not only avoid weaknesses present in existing protocols, but also satisfy new security requirements of wireless networks. While the proposed group key agreement scheme for ad hoc networks achieves great efficiency in computation and communications with a novel key tree construction method. Also using the group key tree structure, our group password-authenticated key exchange protocol provides convenience, scalability and great computation efficiency.

Publications

[1] *Zhiguo Wan*, Bo Zhu, Robert H. Deng, Feng Bao and Akkihebbal L. Ananda, "Efficient Key Tree Construction for Group Key Agreement in Ad Hoc Networks", accepted by IEEE Wireless Communications and Networking Conference (WCNC) 2006.

[2] *Zhiguo Wan*, Robert H. Deng, Feng Bao and Akkihebbal L. Ananda, "Access Control Protocols with Two-layer Architecture for Wireless Networks", submitted to journal of Computer Networks.

[3] *Zhiguo Wan*, Feng Bao, Robert Deng, and Akkihebbal L. Ananda, "Security Analysis on a Conference Scheme for Mobile Communications", accepted for publication in the journal of IEEE Transactions on Wireless Communications.

[4] Kui Ren, Tieyan Li, *Zhiguo Wan*, Feng Bao, Robert H. Deng and Kwangjo Kim, "Highly reliable trust establishment scheme in ad hoc networks", Computer Networks, Volume 45, Issue 6, Pages 687-699, 21 August 2004.

[5] *Zhiguo Wan*, Robert H. Deng, Feng Bao and Akkihebbal L. Ananda, "An Efficient Server-Assisted Group Password-Authenticated Key Exchange Protocol", in submission.

[6] *Zhiguo Wan*, Robert H. Deng, Feng Bao and Akkihebbal L. Ananda, "Anonymous DoS-Resistant Access Control Protocol Using Passwords for Wireless Networks", accepted for publication by IEEE Conference on Local Computer Networks (LCN 2005).

[7] *Zhiguo Wan*, Bo Zhu, Robert H. Deng, Feng Bao and Akkihebbal L. Ananda, "DoS-Resistant Access Control Protocol with Identity Confidentiality for Wireless Networks", IEEE Wireless Communications and Networking Conference 2005 (WCNC'05), New Orleans, 13-17 March, 2005.

[8] Bo Zhu, Guilin Wang, *Zhiguo Wan*, Mohan S. Kankanhalli, Feng Bao, Robert H. Deng, "Providing Robust Certification Services Against Active Attacks in Ad Hoc Networks". Proc. 24th IEEE International Performance Computing and Communications Conference (IPCCC 2005), Phoenix, 7-9 April, 2005.

[9] *Zhiguo Wan* and Shuhong Wang, "Cryptanalysis of Two Password-Authenticated Key Exchange Protocols", in Proceedings of ACISP 2004, pages 164-175, July 13-15,2004, Sydney, Australia, 2004.

[10] Bo Zhu, *Zhiguo Wan*, Mohan S. Kankanhalli, Feng Bao, Robert H. Deng, "Anonymous Secure Routing in Mobile Ad-Hoc Networks", The 29th Annual IEEE Conference on Local Computer Networks (LCN) 2004, Tampa, Florida, U.S.A., 2004.

CHAPTER 1

Introduction

The emergence and fast development of wireless network technologies result in extensive and wide applications in our daily lives. Wireless communications provide great benefits such as flexibility, mobility, portability and low deploy cost for organizations and users. Mobile devices like PDAs, laptops and mobile phones are widely used for various purposes: accessing emails, sharing files, real-time communications etc. While value-added service providers are relying on wireless technologies to provide services to their clients in a more convenient way.

Wireless technologies provide different capabilities that satisfy different users and requirements. Wireless local area networks (WLAN), such as IEEE 802.11, provide short-range, high-speed wireless data connections between mobile devices and nearby access points. Wireless personal area networks (WPAN) like Bluetooth provide a method for interconnecting devices centered around an individual person's workspace. Providing a wireless coverage larger than WLAN, wireless metropolitan area networks (WMAN) enable users to establish wireless connections between multiple locations within a metropolitan area like a city or university campus. Wireless wide area networks (WWAN), such as 2G and 3G systems, provide wireless connections over a large geographic area through the use of multiple antenna sites or satellite systems maintained by wireless service providers. However, a wireless ad hoc networks is a self-organized

infrastructureless network formed by a group of mobile nodes. Such a network provides great convenience and flexibility for users since no infrastructure is required within the network.

Though wireless technologies provide great benefits for users, they also raise concerns on security problems of wireless networks. First of all, openness of radio media leads to more serious security problems in wireless networks besides the same security threats faced by wired networks. In wireless networks, information is transmitted over the open air and anyone can intercept it with suitable devices. As a result, an attacker can easily eavesdrop or launch active attacks against wireless communications. Since there is no physical boundary existing in wireless networks like in wired networks, attackers can easily gain unauthorized access to wireless networks with suitable equipments. What make things worse are resource constraints of wireless networks, which make providing security solutions for wireless networks a very challenging work. Wireless networks usually have a lower bandwidth than wired networks, and mobile devices often have limited computation capability and energy. As a result, it is easy for attackers to mount successful DoS attacks to deplete computation resource and energy of mobile devices. Hence it is important to design efficient security schemes immune to DoS attacks for wireless networks. Mobility of wireless devices also brings privacy problems for roaming users. For a roaming user, his/her movement pattern and location are very important privacy information and should be protected from disclosure. While situations for wireless ad hoc networks are even more complex as infrastructures are not available in such networks. In wireless ad hoc networks, each node can only communicate directly with other nodes within its power range, and some nodes are required to relay packets

on behalf of a source node in order to deliver data to its destination. As a result, security issues in ad hoc networks are more challenging.

1.1 Security Issues in Wireless Networks

Security issues in wireless networks can be considered from three aspects: security requirements, security attacks and security mechanisms. Various security mechanisms are designed to fulfill security requirements so as to counter against different security attacks. Due to characteristics and constraints of wireless networks, wireless networks are facing more security threats than wired counterparts. In this section, we discuss these three aspects of security issues for wireless networks in detail, respectively.

1.1.1 Security Requirements

In traditional networks, *authentication*, *confidentiality* and *integrity* are the three fundamental security requirements studied for tens of years in research. These requirements are also basic research objectives in wireless environments. *Authentication* means that a communication partner can be unambiguously identified during the communication. Sometimes only unilateral authentication is enough for secure communication, while mutual authentication is desired to avoid attacks in most cases. Various authentication protocols are employed to provide mutual authentication for communication networks. *Confidentiality* means that the exchanged information during the communication is not disclosed to unauthorized parties. Encryption, implemented by stream ciphers and block ciphers, is used to achieve confidentiality. *Integrity* ensures consistency of data and detecting unauthorized creation, alteration, or destruction of data. This can

be achieved by using message authentication code (MAC), or message integrity code (MIC). *Non-repudiation* sometimes is also mentioned as a basic security requirement in some applications like billing. This requirement prevents either the sender or the receiver from denying a transmitted message, and digital signature is usually used to provide non-repudiation as well as integrity.

In wireless environments, we also consider the following security requirements. *Availability* ensures legitimate parties are not unduly denied access to resources and services of host networks. This requirement is very important as a network is meaningless if it cannot provide services. To assure availability, security solutions should offer resistance to denial-of-service (DoS) attacks, including memory-DoS, computation-DoS and network bandwidth-DoS attacks. *Access control* requires that only authorized parties can access the wireless network. Fine grained access control, ideally on a per-packet level, should be enforced for wireless networks. *Perfect forward secrecy* is crucial in that it protects previous session keys and confidential messages against compromising of long term secrets, like private keys, passwords. A new requirement introduced by the unique features wireless networks is *anonymity*, which requires the identity of the mobile user should be protected from the network it gains access to. This requirement implies user location privacy and unlinkability between two communications, and protects the user's motion pattern from being disclosed.

At the end, an important requirement on security schemes for wireless networks is *efficiency*. The security solution should be efficient in both computation and communications as mobile devices are usually resource-constrained and the bandwidth is limited in wireless networks.

1.1.2 Security Attacks

Security research in traditional networks has identifies various attacks against communicating parties, and such attacks can be also applied against wireless networks. Generally, these attacks can be divided into two major types: passive attacks and active attacks. Passive attacks do not involve any message alteration, and refer to eavesdropping or traffic analysis. In contrast to passive attacks, active attacks involve some modification or creation of messages during communication. Passive attacks are hard to detect, but they are not as dangerous as active attacks because they do not affect execution of security protocols. Compared to passive attacks, active attacks are much more dangerous and difficult to defend since their active intervention causes much more problems for security protocols. Fortunately, they can be detected by legitimate communication parties.

Common passive attacks mainly include eavesdropping and traffic analysis. Active attacks, however, can be classified into the following categories. *Masquerade* attacks refer to an illegitimate entity pretending to be an authorized entity. While *replay* attacks refer to retransmission of previously captured messages which may result in unauthorized effect. *Message alteration* attacks are to modify messages from an authorized party to produce unauthorized effect. While *Denial of Service* (DoS) attacks aim to degrade performance of networks and prevent normal access to network services and resources. What has been discussed is a general classification of attacks in communication networks, and some attacks may employ much more complex analysis and techniques. For instance, the well-known man-in-the-middle attack is a complex form of masquerade

attack; several parties can also collude to compromise secrets of other parties, which is referred to as the collude attack.

Threat of these attacks has been intensified due to the nature of wireless medium. Attacks against wireless networks can be launched without physical connection to the target networks. For example, attackers can easily eavesdrop or analyze traffic in wireless networks within radio transmission range using a suitable transceiver. Also access to wireless networks is open to attackers as no physical boundary exists. And denial of service attacks are more effective in wireless networks since wireless networks are resource-constrained. Moreover, privacy information like identity and location in wireless networks can be the target of attacks.

1.1.3 Security Mechanisms

Various security mechanisms have been designed to counter against security attacks and satisfy security requirements in wireless networks. Security primitives, like encryption, decryption, signature and one-way hash function, are designed to provide basic cryptographic functions. And based on these security primitives, security protocols have been designed to provide different level of security for communication networks. Among these security protocols , *authentication and key exchange protocols* are the most basic ones that provide basic security services for communicating parties.

Generally, authentication and key exchange protocols can be divided into two groups: two-party and multi-party protocols, the latter of which are also known as group key management protocols. Two-party authentication and key exchange protocols have been well studied in the context of traditional networks, and research results from traditional

networks have been employed in wireless environments. However, existing two-party authentication and key exchange protocols are not satisfactory in security, and they usually fall short of one or more security requirements for wireless networks. Some protocols do not offer client anonymity [6, 18–20], some do not provide perfect forward secrecy [3,4,9,12], while some are unable to offer DoS resistance [3,4,17,19,25]. Moreover, some protocols are even insecure against well-known attacks. It is still a challenging work to design a sound authentication and key exchange protocols that fulfill all the requirements for wireless networks.

With proliferation of group-oriented applications, such as teleconferencing, pay-TV, distributed interactive games, secure group key management protocols for wireless networks are urgently needed to protect group communications. Existing group key management protocols cannot be directly used in wireless networks since they are originally designed for wired networks and differences of wireless networks make them inapplicable in wireless environments. Previous schemes [75, 76, 79] are usually too costly in computation or communications for wireless networks, and hence some efforts have been spent on improving their efficiency to suit requirements of wireless environments. Most group key management schemes exploit a key hierarchy in group key establishment to improve efficiency because of advantages of the hierarchical tree structure. But the hierarchical key tree is usually constructed independent of network topology, which results in inefficiency in communications. Some studies have been conducted to exploit network topology in group key distribution schemes for wireless ad hoc networks [88, 89] and wireless LANs [90]. But similar study has not conducted on group key agreement for wireless ad hoc networks yet.

Group key agreement protocols using only human-memorable passwords are convenient for use and we call them group password-authenticated key exchange protocols. Using human-memorable passwords for authentication and key exchange is most convenient and has been extensively applied in the real world. Although two-party password-authenticated key exchange protocols [98, 99] have been well investigated, password-based group key agreement protocols have not received enough attention and only a few proposals appeared recently [93]. Among these password-based group key agreement protocols, they are either unscalable to large group size or inefficient in computation and communications.

1.2 Thesis Contribution

In this thesis, we studied both two-party and multi-party protocols for authentication and key exchange in wireless environments, and presented several security solutions to achieve authentication and key establishment in wireless networks.

Access control protocols for wireless networks fall into the category of two-party authentication and key exchange protocols, and they are designed to prevent unauthorized access in wireless networks. Access control protocols are important in wireless networks because wireless networks have no physical boundary and can be accessed over the air. Previous access control protocols for wireless networks fail to fulfill some of the security requirements, like anonymity, DoS resistance. In this thesis, we proposed two access control protocols for wireless networks to fulfill all necessary security requirements. The first protocol is based on weak passwords while the second one relies on PKC for au-

thentication and access control. Both protocols are designed to offer user anonymity as well as resistance to DoS attacks for wireless networks.

To avoid inefficiency resulted by constructing the group key tree independent of network topology, we designed a group key agreement scheme in which a key tree is constructed to match the network topology. Such a key tree structure can localize transmission of keying information and hence significantly reduces communication cost of rekeying. We implemented our group key construction scheme on ns-2 and evaluated its performance. Simulation results showed overhead of our scheme is reduced to about 1/4 of other schemes.

This thesis also proposed an efficient and scalable password-based group key agreement protocol for multi-hop wireless networks. In this protocol, each user shares a different human-memorable password with a trusted server, and a group of users from a multi-hop wireless network intend to agree on a group key with the server's assistance. The password-based group key agreement protocol has great efficiency in communications and computation, as a group key tree well-suited for multi-hop networks is specially designed for that purpose. The protocol is also scalable to group size. With this protocol, a group of users can agree on a group key within only 3 flows, and each user needs only $5 + O(\log n)$ exponentiations.

1.3 Thesis Organization

In Chapter 2, we present related work in the area of security in wireless networks. We review access control protocols for wireless LAN first, then we look at the group key

agreement protocols for wireless networks. Finally, we investigate password-based group key agreement protocols.

In Chapter 3, we discuss our two access control protocols for wireless LAN. First we present our password-based protocol for access control in wireless networks. This protocol is designed to avoid security flaws of the so-called Lancaster protocol. Then we discuss the other access control protocol which is based on public key cryptography. We show that both protocols avoid security flaws of previously proposed protocols, and they offer advanced features like client anonymity and DoS resistance.

In Chapter 4, we investigate group key agreement protocols for ad hoc networks. A new group key tree construction approach for ad hoc networks is described and analyzed in detail. We show that how the group key tree in our scheme is constructed from the underlying network topology, and how the constructed key tree can localize rekeying message transmission so as to improve communication efficiency. Finally, we also demonstrate the performance of our scheme by compared with other key tree construction methods.

In Chapter 5, we present our password-based group key agreement protocol, which can be used in multi-hop wireless networks as well as wired networks. We discuss drawbacks of previous password-based group key agreement protocols first, and then propose our protocol. We analyze security of our protocol and show that it is efficient in computation and communications.

In Chapter 6, we conclude the thesis by summarizing the work that have been done. And I also discuss possible future research directions.

CHAPTER 2

Review of Related Work

In this Chapter, we review the literature on security research for wireless networks, including wireless LAN and ad hoc networks. First of all, we give an overview of different types of wireless networks. After that, we review authentication and key exchange protocols for wireless LAN, then we turn to group key agreement protocols for wireless ad hoc networks. Finally, we study password-based key exchange protocols and analyze existing password-based group key agreement protocols.

2.1 Background

2.1.1 Wireless Local Area Networks (WLAN)

Wireless LAN is a kind of local area network that transmits data over the air via high-frequency radio links. In WLAN, wireless base stations (access points) are wired to an Ethernet network and able to transmit messages over an area of several hundred feet through walls and other non-metal barriers. Roaming users can be handed off from one access point to another like a cellular phone system. The main WLAN standards are the IEEE 802.11 standard [33] and HIPERLAN. Other standards like HomeRF, OpenAir are not so influential as 802.11 and HIPERLAN.

IEEE 802.11 is currently the major open standard developed by the working group

11 of the IEEE LAN/MAN Standards Committee (IEEE 802). It consists of a set of different wireless standards: 802.11, 802.11b, 802.11g, 802.11a. IEEE 802.11 is the original standard specifying wireless data transmission, but widespread use of 802.11 networks begins only after 802.11b was ratified. IEEE 802.11b (a.k.a WiFi) is currently the most popular standard. It works at the 2.4GHz band and can transfer data at a speed up to 11 Mbit/s within a range of 30-100 meters. Different from 802.11b working at the 2.4GHz band, IEEE 802.11a operates on the licence-free 5 GHz frequency band. IEEE 802.11a is four times faster than 802.11b, providing a speed up to 54 Mbit/s and a range of 10-100 meters. IEEE 802.11g is the latest standard and is just as fast as 802.11a, but operates on the 2.4 GHz frequency band.

HIPERLAN/1, HIgh PErformance Radio LAN version 1 is an ETSI standard whose goal was to achieve an even higher data rate than 802.11. The standard covers the physical and the MAC part of the Data Link layers like 802.11. Working at the frequency of 5GHz, HIPERLAN/1 has a coverage range of 50 meters, and supports slow mobility of 1.4m/s. HIPERLAN/1 provides transmission throughput of 32 kbit/s for sound, 2 Mbit/s for video, and 10Mbit/s for data. HIPERLAN/2 is designed as a fast wireless connection for many kinds of networks: UMTS back bone network, ATM and IP networks. Also it works as a network at home like HIPERLAN/1. HIPERLAN/2 uses the 5 GHz band and provides a transmission speed up to 54 Mbit/s.

The IEEE 802.11 Wireless LAN Architecture

The 802.11 architecture comprises several components and services that interact to provide station mobility transparent to the higher layers of the network stack.

The wireless LAN station (STA) is the most basic component of the wireless network.

A station is any device that contains the functionality of the 802.11 protocol, and a connection to the wireless media. Typically the 802.11 functions are implemented in the hardware and software of a network interface card (NIC).

A station could be a laptop, a handheld device, or an access point. Stations may be mobile, portable, or stationary and all stations support the 802.11 station services of authentication, de-authentication, privacy, and data delivery. Wireless access points are commonly built into broadband routers, providing both wired and wireless connectivity for a small network.

A typical architecture of wireless LAN is illustrated in Fig. 2.1. The access points are connected by the backbone network to provide wireless access and services for mobile stations. The access point backbone network is connected to the internal network with an access router which performs access control. Within the internal network, RADIUS server, PKI server and other servers provide services like authentication, accounting etc. Before mobile stations can obtain access to the internal network, they usually need to be authenticated and allowed to access by the access router. After mobile stations have access to the internal network, they can access to Internet via the firewall.

2.1.2 Wireless Personal Area Networks (WPAN)

WPAN is a wireless network typically limited to a small cell radius. In an office environment, a WPAN would be used to transfer data between a handheld device and a desktop machine or a printer. For example, a mobile user could download e-mails or Web data into a dual-mode smart phone or PDA and then exchange that data with a machine in the office. In the home, WPANs are expected to provide cable-free connections for

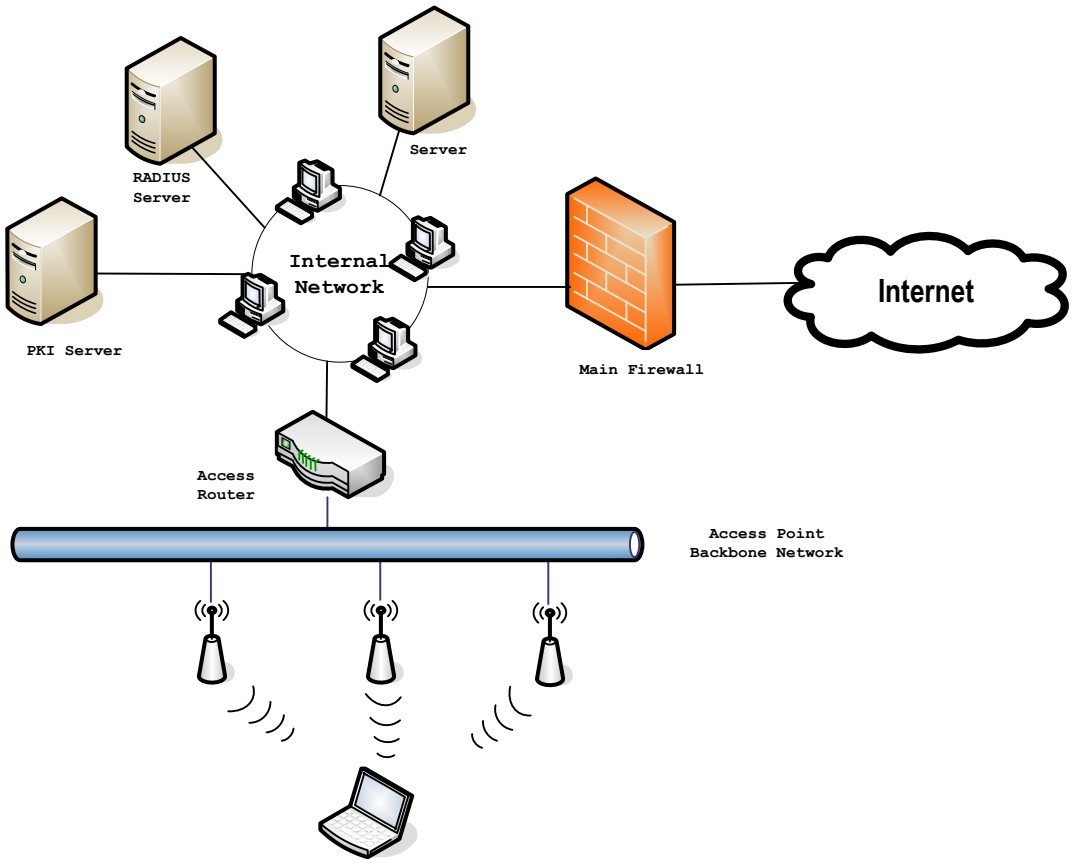


Figure 2.1: A Typical 802.11 Wireless Network Architecture

alarms, appliances and entertainment systems.

Bluetooth is a WPAN technology developed by the Bluetooth Special Interest Group (www.bluetooth.com) founded in 1998 by Ericsson, IBM, Intel, Nokia and Toshiba. Bluetooth provides up to 720 Kbps data transfer within a range of 10 meters and up to 100 meters with a power boost. Bluetooth uses omnidirectional radio waves that can transmit through walls and other non-metal barriers. Bluetooth transmits in the unlicensed 2.4GHz band and uses a frequency hopping spread spectrum technique that changes its signal 1600 times per second.

IEEE 802.15 is a set of standards defined for WPAN. IEEE 802.15.1 defines the lower layers of the Bluetooth specification, and it is approved by the IEEE in 2002. IEEE 802.15.1 is fully compatible with Bluetooth 1.1. IEEE 802.15.3 and 802.15.3a define the high data rate WPAN systems, while 802.15.4 standardizes WPAN for low data rate systems. HIPERPAN is another WPAN standard developed by ETSI in Europe.

Bluetooth WPAN Architecture

Bluetooth communication occurs between a master radio and a slave radio. Bluetooth radios are symmetric in that the same device may operate as a master and also the slave. Two or more radio devices together form ad-hoc networks called piconets. All units within a piconet share the same channel. Each piconet has one master device and one or more slaves. There may be up to seven active slaves at a time within a piconet.

A master is the only one that may initiate a Bluetooth communication link. However, once a link is established, the slave may request a master/slave switch to become the master. Slaves are not allowed to talk to each other directly. All communication occurs within the slave and the master. Slaves within a piconet must also synchronize their internal clocks and frequency hops with that of the master. Each piconet uses a different frequency hopping sequence. Radio devices used Time Division Multiplexing (TDM). A master device in a piconet transmits on even numbered slots and the slaves may transmit on odd numbered slots.

Multiple piconets with overlapping coverage areas form a scatternet. Each piconet may have only one master, but slaves may participate in different piconets on a time-division multiplex basis. A device may be a master in one piconet and a slave in another or a slave in more than one piconet.

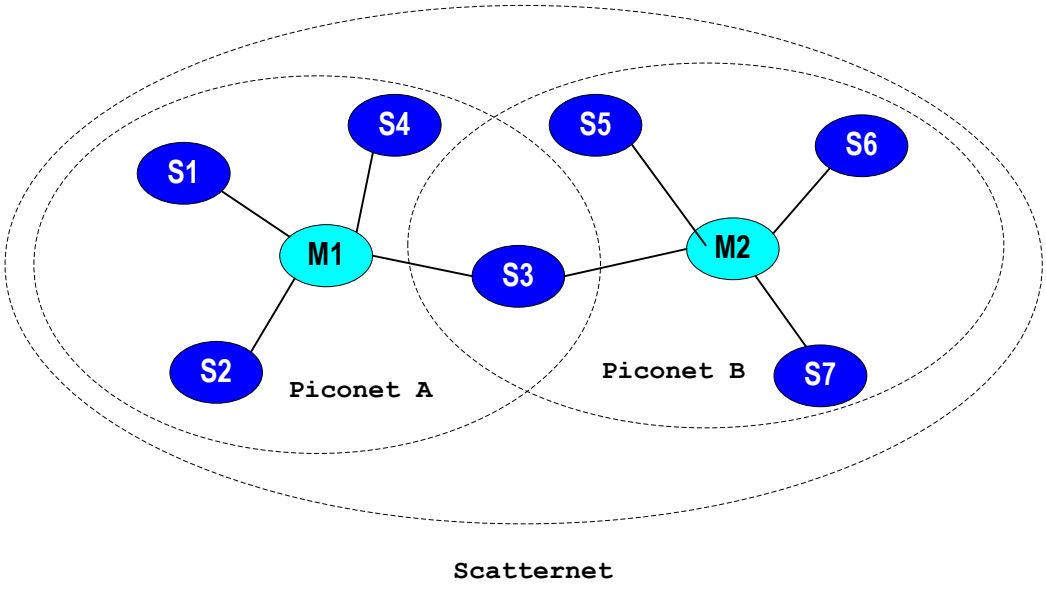


Figure 2.2: Network Topology of Bluetooth WPAN

2.1.3 Wireless Wide Area Networks (WWAN)

Current WWAN technologies include telephony networks like GSM (Global Systems for Mobile Communications), GPRS (General Packet Radio Service), UMTS (Universal Mobile Telecommunications Service) etc. GSM is the widely used 2nd generation cellular network system. This digital cellular system focuses on voice as well as data. But its data rate is too low to be suitable for large amount of data transfer. Developed on the basis of GSM, GPRS introduced packet technology for the first time to support higher data rates, and left voice network unchanged. Even though, it doesn't satisfy the increasing requirement for higher data rate. The 3rd generation (3G) wireless network emerged to offer Internet and Intranet services as well as traditional voice communication service with better performance. UMTS and CDMA2000 are the most important 3G standards specified by 3GPP and 3GPP2, respectively. UMTS uses W-CDMA as the

underlying standard, and represents the European/Japanese answer to the ITU IMT-2000 requirements for 3G Cellular radio systems. UMTS supports up to 1920 kbit/s data transfer rates. CDMA2000 is a 3G mobile telecommunications standard that uses CDMA, and it supports data rate up to 3.1Mb/s. Besides telephony networks, Mobile IPv6 also falls into the WWAN category. Now IETF has standardized Mobile IPv6 with the Internet standard RFC 3775 specifying how the IPv6 Internet operates with mobile computers.

2.1.4 Wireless Metropolitan Area Networks (WMAN)

WMAN is the most important and promising area in wireless networks now. Compared to WLAN, WMAN has a larger coverage area up to a city, and it has a higher data rate up to 70Mb/s. Currently there are several co-existing WMAN standards, including IEEE 802.16, HIPERMAN, and WiBro. The IEEE 802.16 standard, also known as WiMAX, is being supported and promoted by a group of leading vendors of wireless access equipments and telecommunications components. The current 802.16 standard is IEEE 802.16-2004, which only addresses fixed systems. Using the 2-11GHz frequencies which can penetrate walls and other dense objects, 802.16-2004 provides transmission to stationary devices and replaces prior 802.16 and 802.16a specifications. While 802.16e is an extension of 802.16-2004 for mobile use in the 2-6GHz band. It allows people to communicate while walking or riding in cars. In Europe, ETSI developed a similar standard HIPERMAN, which is used mainly within European countries.

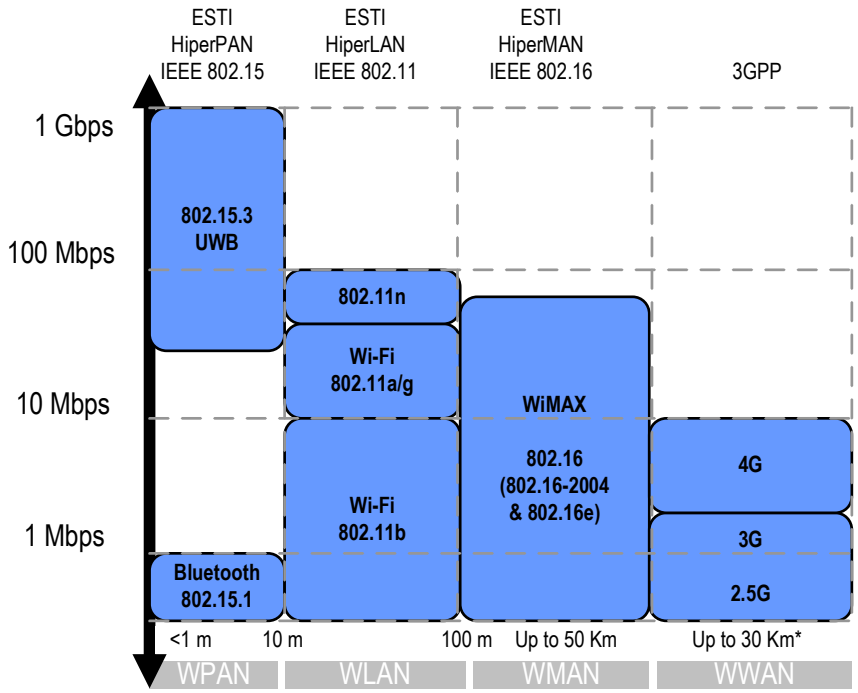


Figure 2.3: Bandwidths and Ranges of Different Wireless Technologies

2.1.5 Mobile Ad hoc Networks

A mobile ad hoc network is an infrastructureless, self-organized wireless network formed by a collection of mobile nodes that can communicate each other via wireless radio. In ad hoc networks, there is no any available infrastructure like routers and servers, and every mobile node needs to serve as a router to forward packets for others besides being a normal node. Every node in ad hoc networks is capable of arbitrary movement, and the network topology is frequently changing.

A number of routing protocols have been proposed for ad hoc networks to facilitate communications within the network. They can be categorized into two groups: table-driven and on-demand routing protocols. Table-driven routing protocols maintain consistent, up-to-date routing information from each node to every other node in the

network. Each node maintains one or more tables to store routing information and propagates topology changes throughout the network. On-demand routing protocols creates route only when the source node has packets to send to the destination. The source node can find the route to the destination node by a route discovery process. Destination-Sequence Distance Vector (DSDV) is a table-driven routing protocol, while Ad hoc On-Demand Vector (AODV) [63] and Dynamic Source Routing (DSR) are on-demand routing protocols.

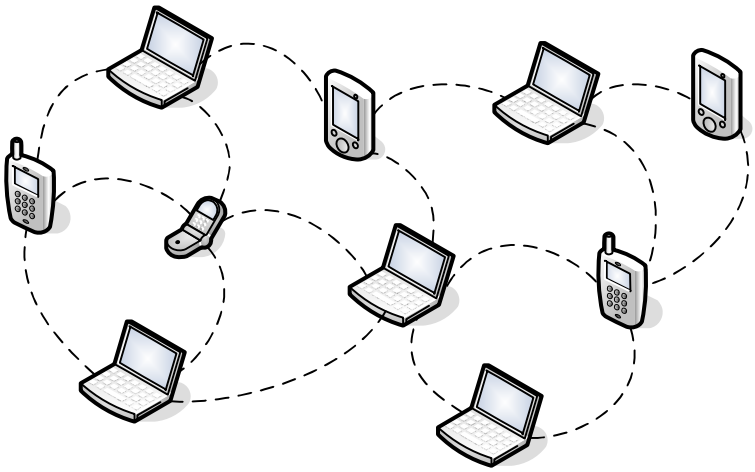


Figure 2.4: A Typical Ad hoc Network

2.2 Authentication and Key Exchange Protocols for Wireless LANs

Due to prevalence of wireless networks, there has been a lot of research focusing on access control and authentication protocols for wireless networks. These protocols are usually designed to authentication and key exchange between a mobile station and a

wireless LAN. Among these protocols, some are based on symmetric cryptosystem, some are based on public key cryptosystems, while some are hybrid cryptosystem based protocols. Unfortunately, existing solutions for wireless networks cannot fulfill all security requirements, and some of them even have serious security flaws.

2.2.1 Protocols Based on Symmetric Cryptosystem

The Wired Equivalent Privacy (WEP) protocol used in the IEEE standard 802.11 [33] relies on symmetric cryptosystem for access control in wireless networks. WEP is intended to protect wireless communications from eavesdropping as well as preventing unauthorized access to wireless networks. It relies on a shared secret between the mobile station and the access point to achieve the aforementioned goals. However, it has been indicated that WEP has serious design flaws that make WEP vulnerable against both passive and active attacks [7, 11]. Moreover, WEP slides over key management problem and leaves it as an open problem for implementation.

To solve the above security problems, IEEE specifies the 802.11i standard [35] to enhance the security of 802.11. In the 802.11i standard, a long term security architecture for 802.11 called the Robust Security Network (RSN) and the Robust Security Network Association (RSNA) are defined for wireless networks. RSNA uses the IEEE 802.1X standard [34], also known as port-based access control protocol, to perform access control, authentication, key management, and key establishment mechanisms. In IEEE 802.1X standard, EAP (Extensible Authentication Protocol), which is a flexible protocol used to carry arbitrary authentication information, is used to carry authentication and key establishment messages. EAP provides flexibility and extensibility for

authentication by defining an independent message exchange layer. Depending on the result of authentication, IEEE 802.1X controls the flow of MAC data units by changing the port status. Actually, IEEE 802.1X is a two-layer access control mechanism in which authentication and access control are implemented at different layers. However, it has been pointed out that the 802.1X protocol is vulnerable to the session hijacking attack and the man-in-the-middle attack [30] if authentication protocols over EAP do not provide strong mutual authentication.

Developed by Cisco, LEAP (Lightweight Extensible Authentication Protocol) [20] over EAP emerges to fill the gap of key management and authentication left by WEP. LEAP is based on symmetric cryptosystem, and it uses a password shared between the client and the server to perform authentication and key exchange. Though LEAP provides a means of mutual authentication and key management for wireless networks, it provides zero resistance against offline dictionary attacks as LEAP can be broken within minutes by dictionary attacks [21].

Basically, protocols relying solely on symmetric cryptosystem are unable to fulfill the requirement of user anonymity as well as perfect forward secrecy. In such protocols, the user needs to disclose his identity so that the server knows which shared secret should be used for authentication and key exchange. And Diffie-Hellman key exchange is not used for key establishment in such protocols, so forward secrecy is not offered.

2.2.2 Password-based Public Key Protocols

Password-based protocols, also known as password-authenticated key exchange (PAKE) protocols, employ weak human-memorable passwords for authentication and key ex-

change. This interesting problem on how to achieve authentication and key exchange using only a human-memorable password is first introduced by Bellovin and Merritt [98], and they also provided a password-authenticated key exchange (PAKE) protocol named the Encrypted Key Exchange (EKE) protocol, and the augmented encrypted key exchange protocol in [99], which is an improvement of the EKE protocol. Since then, it has been a great deal of research effort spent on this subject. According to the number of parties involved in the protocols, PAKE protocols can be divided into two-party and multi-party (group) password-based protocols. In this section, we only discuss two-party password-based protocols, and multi-party password-based protocols are discussed in Section 2.5.3.

IEEE P1363 Standard Working Group has been engaged in standardization on password-based public-key cryptographic protocols. Currently, the working group is studying the PAKE protocols SPEKE [26], SRP [122], PAK [36,116] and AMP [29,109, 110]. Besides these protocols, there are a number of PAKE protocols proposed in the literature. For the PAKE protocols, the most crucial point is their resistance to off-line dictionary attacks (or password guessing attacks). Unfortunately, there have been many attacks against various PAKE protocols in the literature, which in turn shows that these PAKE protocols fail to fulfill the basic requirement.

The PAKE protocol proposed by Zhu *et al.* [123] is specially designed for imbalanced wireless networks. The advantage of this protocol is that one (the mobile node) of the two parties is very lightly computation burdened, which is desirable for mobile nodes in wireless networks. However, as pointed out by Bao [94], the security of this protocol relies on the length of the second party's identity, but not the size of RSA modulo n.

As a result, this PAKE protocol is insecure if the length of the identity is short, which is highly possible in practice.

Several protocols over EAP based on PAKE protocols have also been proposed as IETF drafts, i.e. EAP-PAX [14], EAP-SRP [16], and EAP-SPEKE. The main disadvantages of pure PAKE protocols are their incapability of client identity protection and susceptibility against DoS attacks, and hence they are not suitable for access control in wireless networks. In PAKE protocols, the client requires to disclose his identity to the server so that the server knows which password should be used for authentication. As a result, such protocols cannot provide identity confidentiality for clients. On the other hand, in such protocols the server can only authenticate the client after expensive computation. This causes the protocols susceptible to DoS attacks, since anyone can send requests to launch the server into computational expensive operations. As a result, EAP-SRP and EAP-SPEKE fail to provide user anonymity and resistance against DoS attacks.

Unlike traditional PAKE protocols, the EAP-PAX protocol is a hybrid PAKE protocol where the server holds a certificate, which enables it to provide client identity confidentiality. However, it has several design flaws and cannot meet all requirements of wireless networks. First of all, it is vulnerable to dictionary attacks during its registration phase if the server does not have a certificate. Besides, the protocol replaces the weak password on both the server and the client side with a generated random secret on each update. As a result, the protocol doesn't obtain convenience of using human-memorable passwords in later authentication. Furthermore, the protocol is susceptible to DoS attacks since any part can trick the server into expensive public key

cryptographic decryption.

Due to the disadvantages of traditional PAKE protocols as discussed, it is desirable to design a hybrid PAKE protocol where the server holds a public key certificate like EAP-PAX to offer client identity confidentiality but avoid its weaknesses. Hence we propose a new hybrid PAKE protocol that can offer client identity confidentiality as well as resistance to DoS attacks. Our protocol avoids the disadvantages of traditional PAKE protocols but offers same convenience for users.

2.2.3 PKC-based Authentication Protocols

PKC-based protocols for wireless networks relies on public key cryptosystems for authentication and key exchange. They emphasize on different aspects of wireless network security: some aim at providing user anonymity, some intend to reduce computation complexity, while some focus on non-repudiation. Owing to the asymmetry of PKC cryptosystems, it is easy for them to achieve client identity confidentiality during authentication. But many protocols still fail to fulfill one or more security requirements for wireless networks. However, we propose a PKC-based authentication protocols for access control in wireless networks, which avoids known weaknesses of previous proposals and offers client identity confidentiality and resistance to DoS attacks.

- *The Beller-Chang-Yacobi protocol*

  Designed to offer user anonymity which is important for mobile environments, the Beller-Chang-Yacobi protocol was first proposed by Beller, Chang, and Yacobi [9], and later improved by Carlsen [13] and then by Mu and Varadharajan [32]. The revised BCY protocol employ Modular Square Root public key cryptosystem.

In the protocol, the user's identity and certificate are encrypted with a secret only the server can obtain. However, the BCY protocol does not provide mutual authentication and implicit key confirmation, and anyone can impersonate as the user or the server without being detected. Furthermore, the protocol does not offer perfect forward secrecy as the session key is not generated from two random exponentials by Diffie-Hellman computation.

- *The Aziz-Diffie Protocol*

  The Aziz-Diffie protocol proposed by Aziz and Diffie [4] is designed for privacy protection and authentication in wireless local area networks. But this protocol fails to satisfy several requirements. Since both the client and the server are required to exchange certificates during the protocol, the identities of the client and the server are totally exposed to eavesdroppers. On the other hand, it commits itself in heavy public key computation whenever the server receives a request, which results in susceptibility to DoS attacks. Moreover, the session key is not established with Diffie-Hellman exchange, and hence the protocol does not obtain perfect forward secrecy and compromise of private keys would disclose session keys.

- *The Aydos-Sunar-Kǫc Protocol*

  Aydos, Sunar and Kǫc [6] proposed an authentication and key agreement protocol employing elliptic-curve techniques for wireless environments. However, this protocol fails to fulfill a number of security requirements as mentioned earlier. In the ASK protocol, the server is not authenticated to the user, and user identity anonymity can be easily compromised by comparing users' public keys. The pro-

tocol has no freshness checking and this leads to known-key attacks by replaying corresponding messages. Finally, the protocol has no forward secrecy as session keys can be easily recovered when private keys of both parties are compromised.

- *The Zhou-Lam Protocol*

  The Zhou-Lam protocol by Zhou and Lam [45] was designed for undeniable billing in mobile communications. It is intended for a roaming user to register with a new mobile network and set up a payment mechanism with the new network when the user roams into this network. It is composed of two protocols: the registration protocol and the service request protocol. The registration protocol enables the user and the server to share a common session key with the help of a trusted third party, and the TTP also selects a temporary identity for the user to protect the user's real identity from disclosure. But the protocol does not guarantee authentication of the user to the server though it is not important during registration. The session key is generated only by the trusted third party, which results in lack of forward secrecy.

- *The Boyd-Park Protocol*

  The Boyd-Park protocol [12] is a public key protocol designed for wireless communications. In order for computation efficiency, it does not use Diffie-Hellman key exchange to derive the session key. Instead, the session key is computed as a hash of two random nonces chosen by the user and the server. Therefore, it does not obtain perfect forward secrecy. The benefits of BP protocol include user anonymity and reduced computation complexity.

- *The ASPeCT Protocol*

Developed by the European Commission ACTS project ASPeCT, the ASPeCT protocol [3] is designed for secure communications in personal communication networks. The protocol establishes the session key by Diffie-Hellman computation from the server's private key and the random nonce of the user. In order for user anonymity, the user's identity is encrypted with the established session key in the protocol. Although it provides user identity confidentiality by encrypting user identity, ASPeCT does not obtain perfect forward secrecy because the session key is computed based on the server's private key and the user's random nonce. The protocol also suffers from DoS attacks as the server is required to commit expensive Diffie-Hellman computation after receiving the first message.

- *The IKE and JFK Protocol*

  The Internet Key Exchange protocol (IKEv2) [25] and the Just Fast Keying (JFK) protocol [1] are not originally designed for wireless networks but for wired networks. Due to computation limitation of current mobile devices, IKE and JFK may not get extensive adoption currently. But as the rapid development on processors' processing capability, these protocols can be also employed for wireless networks in the near future.

  The IKE protocol [25] specified in the IETF Internet draft offers identity confidentiality but no resistance to DoS attacks. The JFK protocol [1] specified in another IETF Internet draft gives a solution for providing immunity to DoS attacks and identity confidentiality at the same time. The JFK protocol can resist DoS attacks on exhausting either computation resource or storage resource, and it also can protect the identity of the client from both active and passive attacks.

However, the disadvantage of both protocols is that each user needs a public key certificate for authentication and key exchange, which incurs inconvenience and too heavy a burden for users and organizations.

- *PKC-based Protocols over EAP*

  A set of IETF drafts have defined different security protocols over EAP based on public key cryptosystems: EAP-TLS (Extensible Authentication Protocol - Transport Layer Security) [18], EAP-TTLS (Tunneled Transport Layer Security) [17], PEAP (Protected EAP) [19].

  EAP-TLS was created by Microsoft and accepted by the IETF as RFC 2716: PPP EAP TLS Authentication Protocol. EAP-TLS is the *de facto* standard for authentication in 802.11i wireless LANs. It relies on certificates on both the server and the client side to deliver mutual authentication and secure key exchange, but it fails to protect the client's identity from being disclosed. EAP-TTLS is a proprietary protocol developed by Funk Software and Certicom, while PEAP is developed by Microsoft, Cisco and RSA Security. Both EAP-TTLS and PEAP require only the server certificate to establish a TLS tunnel in stage one, and then authenticate each other in stage two. But they are susceptible to DoS attacks because the server requires to compute a signature upon receiving an authentication request from any entity. Moreover, client identity confidentiality is not provided in PEAP.

Table 2.1: Summary of Weaknesses in Two-Party Authentication and Key Exchange Protocols for Wireless Networks

	Protocols	MA	Anonymity	DoS	PFS	Client-cert
Symmetric key Protocols	WEP	×	×	×	×	
	LEAP		×	×	×	
Password-based Protocols	EKE		×	×		
	Zhu		×	×		
	EAP-PAX			×		
	EAP-SRP		×	×		
	EAP-SPEKE		×	×		
PKC-based Protocols	BCY	×		×	×	×
	AD		×	×	×	×
	ASK	×	×	×	×	×
	ZL	×		×	×	
	BP			×	×	×
	ASPeCT			×		×
	IKE		×	×		×
	JFK					×
	EAP-TLS		×	×		×
	EAP-TTLS			×		
	PEAP		×	×		

2.3 Authentication and Key Management in Wireless PAN

Bluetooth defines several different security levels that can be defined for devices and services. The devices have two trust levels, i.e. trusted and untrusted. The trusted level requires a fixed and trusted relationship and it has unrestricted access to all services, while the untrusted device doesn't have fixed relationship and its access to services is limited. Meanwhile, three different security modes are defined in Bluetooth as follows:

- Mode 1: A non-secure mode in which a device will not initiate any security
- Mode 2: A service level enforced security mode which allows different and flexible access policies for different applications
- Mode 3: A link level enforced security mode in which security procedures are needed.

The security procedures defined in Bluetooth consists of three steps as illustrated in Figure 2.5. In the first step, two units are turned on for the first time and generate their own unit key based on its own address and a randomly generated number. Next, they get in touch for the first time in order to establish a common secret key for following contacts. Subsequently, they generate common secret keys and exchange data under the protection of these keys.

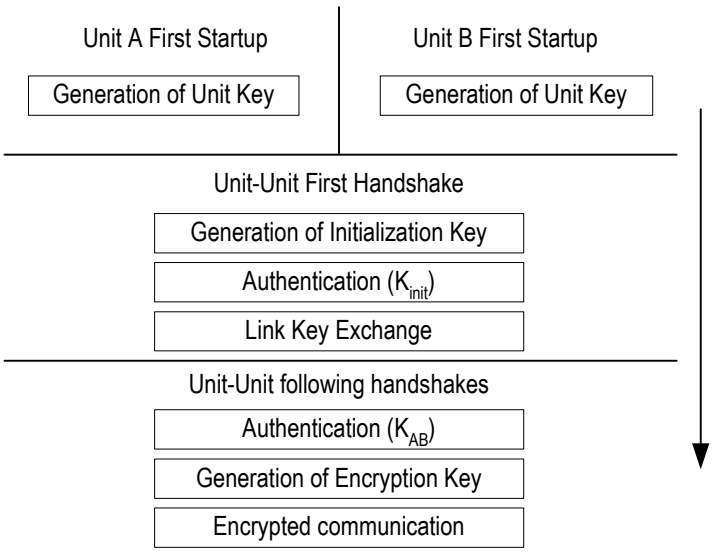


Figure 2.5: Bluetooth Security Overview

2.3.1 Key Management

There are several kinds of keys in Bluetooth system to ensure secure transmission. The most important key is the *link key*, which is used between two Bluetooth devices for authentication purpose. After authentication, an encryption key is derived from the link key and is used to secure the exchanged data between the two devices.

In Bluetooth, four link keys are generated during the security procedures:

- *Unit Keys K_A and K_B*:

  The device generates its own unit key (say K_A and K_B for the two devices) based on its own BD_ADDR (which is a unique address link the MAC address for Ethernet cards) and a randomly generated number.

- *Initialization Key K_{init}*:

  In *Unit-Unit First Handshake*, an initialization key is generated between the two units. The generation of this initialization key takes the Personal Identification Number (PIN) and a random number as parameters.

- *Combination key K_{AB}*:

  Combination key K_{AB} is derived from two units A and B using link key exchange. This key is generated for each pair of devices and is used when more security is needed.

- *Master key K_{master}*:

  Master key K_{master} is used when the master device wants to transmit to several devices at ones. It overrides the current link key only for one session.

After successful authentication, the encryption could be enabled and an encryption key K_C is generated for encrypting communication data. The relationship between link keys and encryption key is illustrated in Figure 2.6.

2.3.2 Authentication

Authentication in Bluetooth employs the challenge-response scheme. It starts by issuing a challenge to another device and it has to then send a response to that challenge which is based on the challenge, it's BD_ADDR and link key shared between them. After

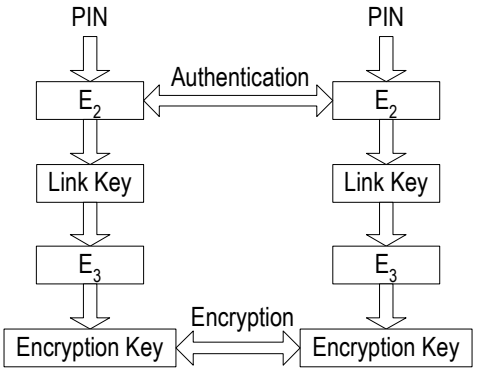


Figure 2.6: Bluetooth Key Management

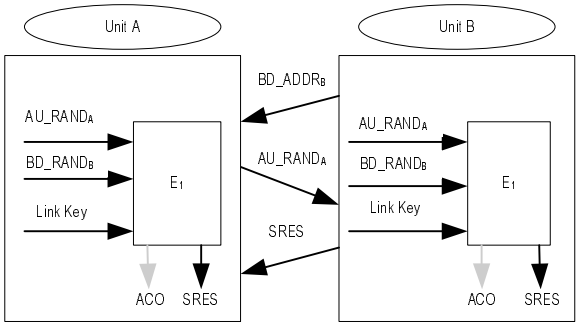


Figure 2.7: Bluetooth Authentication

authentication, encryption may be used to communicate. The authentication procedure is illustrated in the Figure 2.7. During the process of Bluetooth authentication, unit B sends its address to A first, and A replies with a random number AU_RAND_A. Then B takes its address, AU_RAND_A, and the link key as the input to encryption function E_1 to obtain a response $SRES$ and ACO (authenticated ciphering offset). $SRES$ is then sent to A for authentication.

2.3.3 Security Limitations of Bluetooth

Bluetooth has shown some weakness and limitation in its security specification. Firstly, it suffers from eavesdropping because the PIN has a small key space usually. The PIN is usually a 4-digit one, which limits the key space to $10,000$ different values. Hence the attacker can guess all PINs and the correctness of each guess is verified by performing the second step of the initialization protocol. Secondly, only the device not the user is authenticated when the PIN code is stored on the bluetooth device. Therefore, if the device is lost or stolen, any other person can get authenticated with the device. Also, Bluetooth doesn't define authorization separately for each service.

2.4 Authentication and Key Management in Wireless WAN

Universal Mobile Telecommunications Service (UMTS) [62] has emerged as the full-fledged 3G wireless standards to support both the radio and network functions based on the IMT-2000 framework. It is designed to deliver wireless services with better performance than the 2G counterpart. Besides offering traditional voice communication, 3G data capability offers Internet and Intranet services for multimedia application, high-speed business transaction and telemetry.

2.4.1 Security Mechanisms of UMTS

The security functions of UMTS are based on what have been implemented in GSM. Some of the security functions have been added and some existing have been improved. Encryption algorithm is stronger and included in base station (Node-B) to radio network controller (RNC) interface , the application of authentication algorithms is stricter and subscriber confidentially is tighter.

UMTS specification has five security feature groups [62]:

1. *Network access security (I)*: the set of security features that provide users with secure access to 3G services, and which in particular protect against attacks on the (radio) access link. It includes user identity confidentiality, entity authentication, confidentiality, data integrity and mobile equipment identification.
2. *Network domain security (II)*: the set of security features that enable nodes in the provider domain to securely exchange signalling data, and protect against attacks on the wireline network;

3. *User domain security (III)*: the set of security features that secure access to mobile stations
4. *Application domain security (IV)*: the set of security features that enable applications in the user and in the provider domain to securely exchange messages.
5. *Visibility and configurability of security (V)*: the set of features that enables the user to inform himself whether a security feature is in operation or not and whether the use and provision of services should depend on the security feature.

Figure 2.8 provides an overview of the complete security architecture of UMTS.

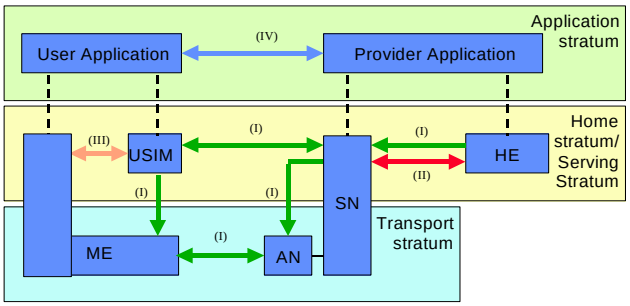


Figure 2.8: UMTS Security Architecture
TE—Terminal Equipment; USIM—User Service Identity Module; SN—Serving Network; HN—Home Network; MT—Mobile Termination; AN—Access Network

2.4.2 Authentication and Key Management

We focus on authentication and key management mechanism of UMTS specification. Authentication in UMTS adopts a challenge-response scheme similar to that of GSM. After the mutual authentication, a new pair of cipher and integrity keys between the VLR/SGSN and the USIM is established. The mechanism employed in UMTS to achieve authentication and key management is shown in the following figure 2.9.

Three entities are involved in the mutual authentication: Home Environment HE

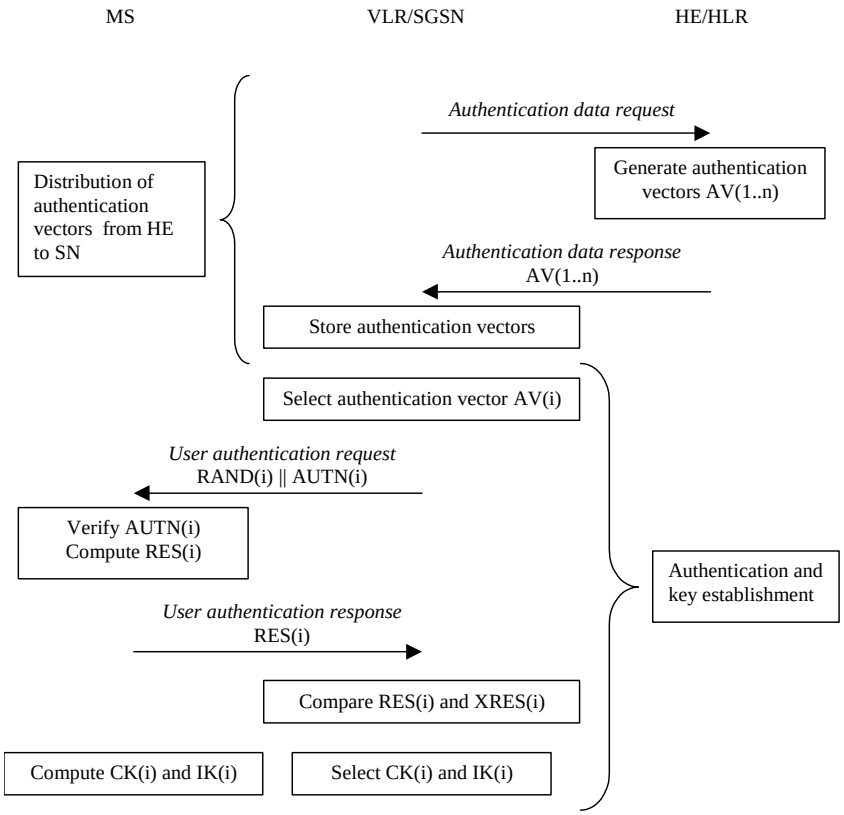


Figure 2.9: UMTS Authentication and Key Management

(AuC), Serving Network SN (VLR/SGSN) and Mobile Station MS (USIM). A pre-shared master key K between MS and HE is the basis of the authentication. A successful authentication and key agreement procedure is as follows.

1. VLR/SGSN requests an Authentication Vector (AV) from the HLR.
2. HLR computes one or more AV. This is done by means of the authentication algorithms and the users private secret key K. (K is only found in the HLR and on the USIM).
3. HLR responds by sending n Authentication Vectors back to the requestor.
4. VLR/SGSN challenges the user/USIM by sending RAND and an Authentication

token (AUTN).

5. The user/USIM processes the AUTN. The AUTN contains a sequence number (SEQ), a message authentication code (MAC-A) and an authentication management field (AMF). With the aid of the private secret key K, the user is able to verify that the received challenge data could only have been constructed by someone who had access to the same secret key K. The user/USIM will also verify that the AV has not expired by checking the sequence number (SEQ). Provided that the network can be authenticated and that the AV is still valid (fresh), the user/USIM proceeds to generate the confidentiality key (CK), the integrity key (IK) and the response (RES).
6. The user/USIM responds with the RES to the network.
7. The VLR/SGSN verifies that response is correct by comparing the expected response (XRES) from the AV with the response (RES) received from the user/USIM.

After having established session keys (CK and IK) through the UMTS AKA procedure, confidentiality and integrity protection services can be initiated. The confidentiality protection applies to both user data and the associated system signalling data.

2.4.3 Security Limitations of UMTS

Improved from GSM, UMTS provides much stronger security features than GSM. UMTS provides security against using false base stations with mutual authentication. In UMTS, encryption is extended from air interface only to include base station to radio network controller connection. Security data in the UMTS network will be protected in data storages and while transmitting ciphering keys and authentication data in the system.

UMTS also provides mechanism for upgrading security features. But it also has a number of security weaknesses.

Identity confidentiality is not absolute with authentication and key management of UMTS. If the user is registering for the first time in the serving network, the permanent user identity (IMSI) is sent in plaintext. An eavesdropper can intercept this message and compromise the user's identity confidentiality.

Due to expensive computation cost of public key cryptography, UMTS does not employ PKC in authentication and key management. This leads to the drawback of lacking perfect forward secrecy in the authentication protocol. Once the pre-shared master secret key is compromised, previous communication content is also disclosed.

Since the 3G systems have been deployed only for a relatively short time, their security still needs more time to be tested. But its security features discussed above clearly guarantee higher level of security than that of previous cellular telecommunication systems.

2.5 Group Key Management Schemes for Wireless Networks

As group-oriented applications are becoming more and more important, group key management schemes are required to keep communications among group members secure. According to mechanisms by which the group key is generated, existing group key management schemes can be classified into two categories: group key distribution schemes and group key agreement schemes.

Group key distribution schemes rely on a centralized server or a group leader to generate and distribute keys to group members. Since the entity responsible for key generation and distribution could likely become the single point of failure as well as the performance bottleneck, group key distribution schemes are not preferable in ad hoc group communications. On the other hand, because the group key is selected by the centralized server, the security of the group relies heavily on how good the group key is selected by the server or the leader.

In group key agreement schemes, a group key is constructed from all the shares contributed by every group member; therefore, such schemes avoid reliance on a single entity for key generation. Group key agreement schemes can be subdivided into centralized schemes and distributed schemes. Centralized schemes use a centralized server for group key management, while distributed group key agreement schemes do not require the existence of a centralized server.

2.5.1 Group Key Distribution

Many conference distribution schemes, a variant of group key distribution, designed for wireless communication systems have been shown to be insecure. The key distribution scheme proposed by Tatebayashi *et al.* [59] is attacked in [56] because the scheme uses a low exponent RSA. Based on the conference key distribution protocol [48] for the wireless environment, a conference scheme with dynamic participation was proposed in [49], and another one employing self encryption was proposed in [50]. But the former [49] has been shown to be insecure in [55], and even worse, Bao showed both [49] and [50] are vulnerable to a colluding attack and a passive attack in [47].

The *Logical Key Hierarchy*(LKH) proposed by Wong *et al.* [71] and Wallner *et al.* [73] are the first to introduce the tree structure into the group key management. In the logical tree structure of LKH, each leaf node corresponds to a group member while the root corresponds to the group key known only to the group members. An improvement of LKH which is known as LKH+ [80] halves the size of rekeying messages. The *One-way Function Tree*(OFT) proposed by McGrew and Sherman [69] improves the hierarchical tree approach furthermore. In OFT, the key of the parent is derived from the keys of its two children, and hence it reduces the size of the rekeying messages to half of the LKH.

Aware of inefficiency of previous group key management schemes in communications, some solutions for group key distribution in wireless networks and ad hoc networks focus on achieving better performance in communications. Lazos and Poovendran [88, 89] proposed a location-aware and a routing-aware key distribution scheme to address the problem of efficiently securing multicast communication for ad hoc networks. The location-aware scheme employs an iterative clustering algorithm to construct a key tree on which nodes located close to each other are clustered together. A variant of K-means algorithm for creating appropriate clusters is introduced to divide group members into clusters. The location-aware scheme requires an additional GPS system to obtain each node's coordinates, and it requires complex computation to obtain the key tree. Moreover, the resulting key tree may not be optimal since it does not consider the position of the group controller. The routing-aware key distribution scheme can build a routing-aware key tree with low complexity, but the efficiency of the resulting key tree is far from that of the optimal key tree. Another group key distribution scheme proposed in [87] uses codewords to represent paths and group nodes based on

the length of the common path, which is derived from hamming distance of codewords. Sun *et al.* [90] proposed a scalable multicast key management for wireless networks. Noticing the localization of rekeying messages transmission, they proposed to construct a topology-matching key management tree to yield efficiency in communications. The topology-matching key management tree they build has a three-layer hierarchy consisting of mobile nodes, base stations and supervisor hosts. Since it is specially designed for wireless LANs, the scheme is not well-suited for ad hoc networks.

2.5.2 Group Key Agreement

The earliest research effort on group key agreement is due to Ingemarsson *et al.* [51]. In their protocol, the group members are arranged in a logical ring, and the protocol executes in $(n-1)$ rounds. In each round, every group member raises the received intermediate key value to the power of its own exponent and forwards the result to its next member. Burmester and Desmedt proposed another group key agreement protocol in [102]. The BD protocol takes only three rounds and two modular exponentiations per member to generate a group key. However, at least half of the members need to change their session random on every membership event, and it requires $2n$ broadcast messages which is expensive on a wide area networks. The group Diffie-Hellman key exchange protocol (GDH) [79] is extended from two-party Diffie-Hellman key exchange. GDH provides fully contributory authenticated key agreement, but it is computation-intensive as it requires $O(n)$ cryptographic operations upon each key change.

Based on the tree structure, Kim *et al.* proposed the tree-based group Diffie-Hellman (TGDH) key agreement protocol [75, 77]. TGDH, which provides great computation

and communication efficiency, combines the key tree technique and the Diffie-Hellman exchange in a fully distributed way. Another group key agreement protocol named STR [76] is an extreme form of TGDH since it is based on a completely imbalanced tree structure. The authors of STR realized the importance of communication efficiency in the long delay networks and STR is aimed at reducing the communication rounds. But the key tree of STR is also constructed without any relation to the network topology, STR reduces the amount of key information needed for group key agreement at cost of more computation efforts. Of these group key agreement protocols, TGDH has been shown to exhibit the best performance in both WAN and LAN settings [81]. Although tree-based group key management schemes provide great efficiency in computation, the key trees constructed in these schemes are usually independent of the network topology, which makes these schemes inefficient in communication.

Therefore, we propose in this thesis an efficient group key agreement scheme with a novel key tree construction approach for wireless ad hoc networks. We present a tree-transformation algorithm which constructs an efficient key tree from the underlying multicast tree of the ad hoc network. The algorithm transforms a multicast tree into a binary key tree which localizes keying information transmission in group key agreement. Our group key agreement scheme based on this key tree largely reduces the communication cost while keeping the computation cost at a very low level.

2.5.3 Multi-party Password-based Protocols

Though multi-party password-based protocols (a.k.a. group PAKE protocols) offer great convenience for group-oriented applications, they only received very modest research

effort.

In group password-based protocols, either the whole group share a single password, or each client in the group shares an independent password with a trusted server. The single-password setting is not preferable in real applications for several reasons. First, if a client in the group leaves or the password of a client is compromised, the shared password has to be updated, which could be a very expensive process. Moreover, compromise of any client leads to breakdown of the entire system. Secondly, individual client identification is impossible in this setting. As a result, no one is able to distinguish one client from another, and it is impossible for a subgroup of the group to securely establish a session key and have secure communications. While the independent-password setting avoids the above problems and reflects more accurately what is happening in the real world.

Under the single-password setting, Asokan and Ginzboorg [93] proposed a group PAKE protocol for ad hoc networks. Bresson *et al.* [101] proposed another group PAKE protocol and proved its security formally. While for the independent-password setting, only a group PAKE protocol EKE-U [95] was proposed under this setting recently. Although this protocol has a formal security proof, it is very inefficient in computation and communications. The protocol requires $O(n)$ exponentiations on client side and $O(n^2)$ exponentiations on server side. Moreover, each client requires to run a one-round TF protocol with the server, which would incur too much communication overhead in wireless networks.

Group PAKE protocols under the independent-password setting need more careful treatment since they suffer from attacks which are not present under the single pass-

word setting, such as attacks initiated by some colluding legitimate clients against other clients' passwords. Not only should passwords be resistant to outsider attacks, but they should be secure against insider attacks. Our group PAKE protocol to be presented in this thesis, however, is designed to counter against all possible attacks under the independent-password setting. Furthermore, our proposed protocol has much better efficiency in terms of communications and computation as it exploits a group key tree structure for group key agreement. This protocol is well-suited for multi-hop wireless networks with trusted servers provided, like wireless mesh networks.

CHAPTER 3

Authentication and Key Exchange in Wireless LANs

3.1 Introduction

The demand for access to wireless networks in public places, such as airport lounges, college campuses and city centers, has surged dramatically over the recent few years. This is mainly due to the growing popularity of mobile devices and the increasing pervasiveness of wireless technologies, such as IEEE 802.11, HomeRF, HIPERLAN/2 and Bluetooth. A major concern in wireless networking is security and in particular network access control. Since deployment of wireless network technologies in public places bears the danger of unauthorized users gaining access to network services, it is extremely important to be able to restrict access to the network only to authorized users. Therefore, secure user authentication and authorization, and a reliable access control mechanism are vital for wireless networks.

Two protocols which have similar two-layer access control architectures for wireless networks in public places have been recently proposed in the literature. The first access control protocol, called the Lancaster protocol [23, 39], is designed for a wireless overlay network around Lancaster city, UK; it employs user password for authentication and enforces access control at the IP layer. However, the design flaws of the Lancaster

protocol make it vulnerable to various attacks.

The second access control protocol, refereed to as the Stanford protocol [22], aims to overcome several security deficiencies in 802.1X [34] and to provide access control in both wireless and wired networks; it uses public key cryptosystems (PKC) for authentication and performs access control at the link layer. Although this protocol is supposed to resist DoS attacks, it is unfortunately susceptible to DoS attacks as well as to other types of attacks.

The models of these two protocols are representative for access control in overlay wireless access networks. Both models adopt a two-layer access control architecture, in which a higher layer is responsible for authentication and key agreement while a lower layer is responsible for per-packet access control. The difference of the two models is the different credentials on which the two protocols are based. The Lancaster protocol relies on a secret password shared between the client and the server, while the Stanford protocol depends on the certificates of the client and the server for authentication and access control. Password-based and certificate-based access control protocols are two different approaches with different advantages: the former approach is more convenient and user-friendly, while the latter can offer more security features like DoS resistance and identity anonymity.

In this chapter, we discuss in detail the flaws of these two access control protocols and propose two secure protocols in place of the Lancaster protocol and the Stanford protocol, respectively. The first protocol proposed is a password-based protocol. It provides great user-friendliness and convenience for clients since no PKC infrastructure but weak passwords are required on the client side. Our second protocol, which is a PKC-

based protocol, is designed for wireless networks where PKC infrastructure is available. Other than mutual authentication and secure key exchange, our PKC-based protocol provides more protection for wireless networks: it employs the cookie mechanism to defend against DoS attacks, while protects clients' identities from disclosure to offer identity confidentiality for clients. Compared with other PKC-based protocols, our protocol provides better resistance to DoS attacks while avoiding their weaknesses.

3.2 Password-based Authentication and Key Exchange for Wireless Networks

In this section, we first review the Lancaster access control architecture considered in [23, 39], and present the Lancaster protocol designed for the architecture; we discuss its security weaknesses and then introduce our protocol for the access control architecture. Both the Lancaster protocol and our protocol use user password for authentication and key exchange. Password based scheme is still the dominate approach for user authentication, this is particularly true for mobile users who may regularly employ a number of different devices as well as many different points of network access.

3.2.1 The Lancaster Access Control Architecture

The Lancaster access control architecture [23, 39] is for publicly accessible wireless overlay networks. It is designed to address the problem of ubiquitous Internet service provisioning within the city of Lancaster. The Lancaster access control architecture, shown in Fig. 3.1, is composed of a number of base stations connected with access routers,

which in turn connect to the authentication server via a gateway. This approach is in fact a two-layer architecture where the authentication server is responsible for client authentication and authorization, while access routers control access to the protected network based on the result of the authentication and authorization.

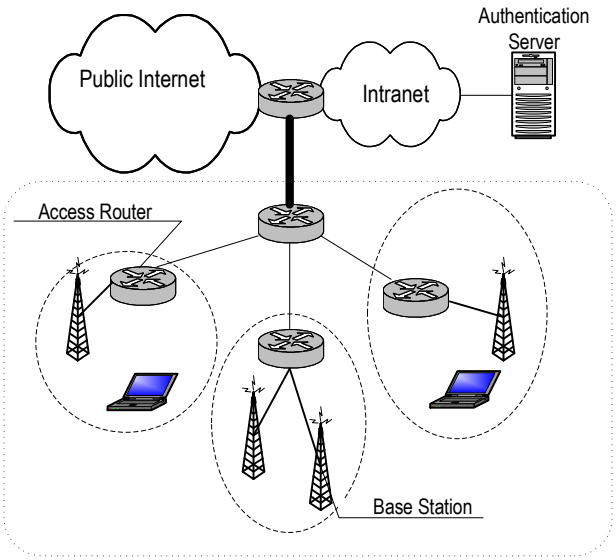


Figure 3.1: The Lancaster Access Control Architecture.

Under the Lancaster architecture, it is assumed that a client holds a secret password corresponding to the client's identity, and shares the password with the authentication server. The authentication server has a private and public key pair and the public key is made known to all the clients. In addition, the access routers have their private and public key pairs with the public keys made available to the authentication server.

3.2.2 Security Requirements

Before proceeding with the description of password based authentication and access control protocols, we list below a number of security requirements for such protocols.

- *Mutual Authentication:* Authentication of the client to the authentication server and authentication of the server to the client. The network want to be sure that it is communicating with a genuine client; otherwise there is a danger that spurious client will be able to fraudulently gain a level of service without ever intending to pay for the service. Authentication of the authentication server to the client is also necessary in order to prevent a type of man-in-the-middle attack as described in [30].
- *Key Authentication and Key Confirmation:* Key authentication requires that only the legitimate participants in the protocol but no other entity possess the agreed secret key, while key confirmation means that both parties in the protocol can be assured that both of them derive the same secret key.
- *Key Freshness:* The protocol should guarantee that the newly constructed session key shared between the two parties has never been used in previous sessions. Otherwise, a compromised old session key can lead to disclosure of subsequent exchanges between the two parties.
- *Perfect Forward Secrecy:* Previous session keys and confidential messages should be protected against compromise of the passwords and other long-term secrets.
- *Secure Against Dictionary Attacks:* Since passwords must be memorable, a secure password based protocol should resist brute-force guessing, or dictionary attacks.
- *Access Control:* Only authorized clients can obtain access to the wireless network. To protect from the parking lot attacks [7], fine grained access control, ideally on a per packet level, should be enforced.

3.2.3 The Lancaster Protocol and Its Security Analysis

The Lancaster protocol, as illustrated in Fig. 3.2, consists of three messages.

1. To access the network, a client initiates the process by sending an authentication request to the authentication server via an access router:

$$E_S(MAC_C, IP_C, K, Username, Password) \tag{3.1}$$

where MAC_C, IP_C, $Username$ and $Password$ are the client's MAC address, IP address, username and password, respectively, and K is a secret session key generated by the client. This request is encrypted with the public key of the authentication server.

2. Upon receiving the authentication request, the authentication server decrypts it using its private key. It checks the received password with the one in its database. If the two passwords match, the client is considered authentic. The authentication server then generates an authentication token $Token$, encrypts it using the session key K and sends the ciphertext to the client:

$$e_K(Token) \tag{3.2}$$

3. Next, the authentication server encrypts the client's MAC address, IP address, the access token and the session key with the access router's public key, and sends

the result to the access router:

$$E_{AR}(K, Token, MAC_C, IP_C) \tag{3.3}$$

The access router decrypts this message and stores MAC_C, IP_C, $Token$ and K into an access control list (ACL).

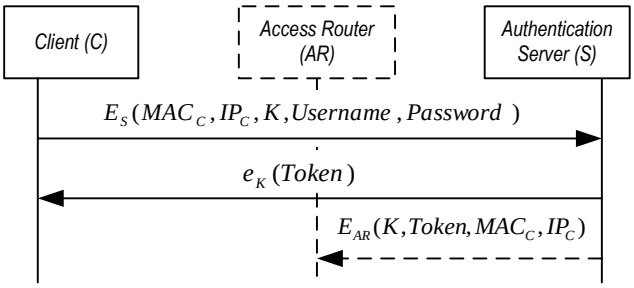


Figure 3.2: The Lancaster Protocol.

When the client sends a packet to the network, it includes an access control extension header in the IP packet as illustrated in Fig. 3.3. This header contains the access token and a checksum both encrypted with the session key K using a symmetric key cipher. In Fig. 3.3, V denotes the protocol version, T denotes the type of services, and Res denotes the reserved bits.

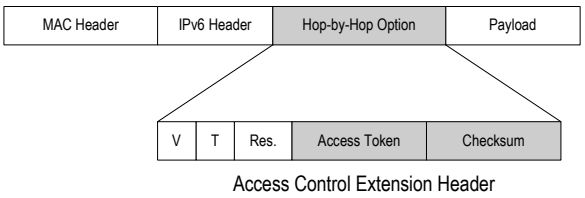


Figure 3.3: The Packet Header Format in the Lancaster Protocol.

The access router checks packets from the clients (i.e., the wireless network) for purpose of access control. When a packet is received from the wireless network, the

access router looks up the MAC address in the ACL. If an entry for the client device exists, the access router verifies the IP source address. In the case of a match, it decrypts the access token and the checksum using the session key and validates its content against the ACL. When successful, the access control extension header is stripped off and the packet is passed on. Packets that fails any of those tests are dropped. One exception to this rule is that when a client is first seen in a cell, it is allowed to contact certain well-known IP addresses; this allows clients to initially communicate with the authentication server.

The Lancaster protocol is simple in design, with only two message exchanges between a client and the authentication server and one message sent from the authentication server to an access router. Unfortunately, it has many serious security flaws which make it vulnerable to various attacks.

First of all, the protocol does not follow the well-known "challenge-response" principle. Specifically, message (3.1) is not sent in response to any challenge from the authentication server. This makes the protocol subject to replay attacks. Obviously, message (3.1) can be replayed by anyone to the authentication server and the server will accept the message and believes the sender as authentic. This design flaw can lead to severe vulnerabilities as discussed below.

If the attacker is able to compromise just one session key K, he can replay message (3.1) to request authentication, and then with the knowledge of the session key the attacker can obtain the access token by decrypting message (3.2). Employing techniques of IP spoofing and MAC spoofing, the attacker can then gain full access to the wireless network with the access token. This attack can be easily launched by modifying outgoing

packets' IP header and MAC header. The attacker can perform this attack any time he wishes to with the knowledge of just one session key even he does not know the client password at all.

The replay attack also leads to DoS attacks for the system. An attacker can collect a large number of authentication requests (i.e. message (3.1)) from different clients, and replays them to the authentication server during a short period of time. Every authentication request will engage the authentication server into expensive computation, including two public key operations and a symmetric key operation. What is more serious is that this attack would prevent the valid clients from accessing the wireless network. After the attack replays a previous authentication request of a client to the server, the server sends the third message to the access router to update the ACL information of the client. Hence the valid client cannot access the wireless network because he does not know which session key and access token the access router uses for performing access control. This attack is serious since an attacker can prevent many valid clients from access the wireless network without any knowledge. Besides, this also leads to a severe computation-DoS attack against the server.

Secondly, since the password space is normally small, the attacker can perform dictionary attacks against message (3.1) if any session key is exposed to the attacker. And this attack can disclose the client's password. The attacker can simply selects a trial password $Password'$ from the password dictionary exhaustively, and computes an encrypted message $E_S(MAC_C, IP_C, K, Username, Password')$ to compare with the intercepted message (3.1). But this dictionary attack can be prevented by using a PKCS#1 public key encryption algorithm.

Thirdly, the protocol does not provide key confirmation for both parties. Neither party is ensured that the other party shares the same secret session key. In message (3.2), $Token$ is selected and encrypted with K by the server, hence the client is unable to confirm that they share the same session key K. Also, key freshness is not guaranteed in the protocol. There is no mechanism to prevent reuse of old session keys. If the client reuses a previously used session key, the server will simply accept this old key. This leads to the failure of the protocol when only one session key is compromised.

An even more serious security loophole is message (3.3) which carries no authentication information about its sender and the message itself has no freshness protection. As a result, an attacker just generates a key K and an access token; he then encrypts the key, the token, a mobile device's MAC and IP addresses using the access router's public key to obtain $E_{AR}(K, Token, MAC_C, IP_C)$, sends the ciphertext to the access router, and starts to access the network service with the mobile device. However, this attack can be easily prevented by using a signature [23] or a secure tunnel as in our password based protocol.

Moreover, the technical details of the access control extension header shown in Fig. 3.3 is not clearly spelled out in [23, 39]. Since the client and the access router share a secret session key K, the use of the access token is not clear. We note that the combined use of checksum and symmetric key encryption as in the Lancaster protocol is dangerous if not designed carefully [10], [11]. The description on the construction of the access control extension header in [23, 39] does not provide enough technical details for us to make creditable analysis.

3.2.4 Our Protocol for the Lancaster Architecture

As discussed earlier, the intrinsic characteristics of PAKE protocols lead to their incapability of providing client identity protection and susceptibility against DoS attacks. In this section, we propose a secure protocol to meet all the requirements.

Before the protocol starts, the client and the server agree on a set of security parameters: a multiplicative group $\mathbb{Z}_p^*$, its subgroup $\mathbb{G}_{p,q}$ of order q and a generator g of $\mathbb{G}_{p,q}$, where p, q are large prime numbers. Specifically, p is selected as a safe prime or a secure prime, which means that $p = 2q + 1$ or $p = 2qr + 1$ where all the factors of r are comparable to q (otherwise the scheme would be insecure against Pohlig-Hellman attack.). Assuming that discrete logarithm problem over $\mathbb{G}_{p,q}$ is hard.

Before a client can access network services, it performs the following authentication and key agreement with the authentication server. The protocol message exchanges are illustrated in Fig 3.4.

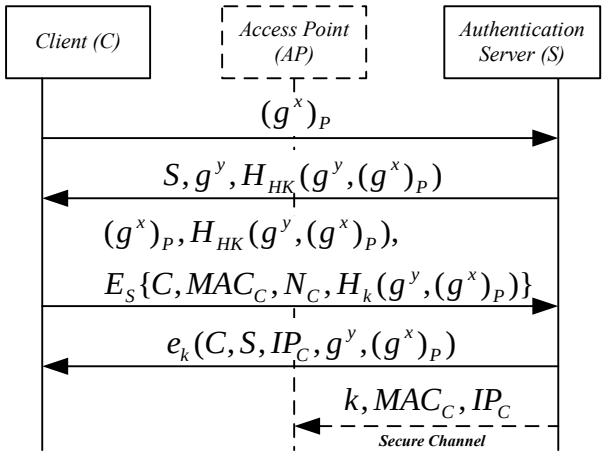


Figure 3.4: Our Anonymous DoS-Resistant Access Control Protocol.

1. The client C chooses a random number $x \in_R Z_q$ and computes the exponential

$(g^x)_P$ encrypted with its password. Then he sends

$$(g^x)_P \tag{3.4}$$

to the server, where C is the identity of the client. Since the exponential g^x is randomly generated, dictionary attacks are not applicable to disclose the client's password.

2. After the server receives the first message, he chooses a random number $y \in_R Z_q$ and the exponential g^y. He computes a hash $H_{HK}(g^y, (g^x)_P)$ with a hash key HK private to the server only. Then the server sends

$$S, g^y, H_{HK}(g^y, (g^x)_P) \tag{3.5}$$

to the client.

In this step, the server should avoid expensive computation in order to resist DoS attacks, because the server cannot determine whether the client is valid. Actually the computation cost of the server includes an exponentiation and a hash computation in this message. But the server can generate the random exponential beforehand or periodically, so that the computation cost of the server is only a light-weight hash computation. In this message, HK is a hash key known only by the server, and it is updated frequently to prevent accidental disclosure. As a result, the hash $H_{HK}(g^y, (g^x)_P)$ can serve as an authenticator and a cookie that would be sent back by the client in the next message.

3. After the client receives the above message, he computes the session key $k =$

$H(C|S|g^{xy})$ and then a hash $H_k(g^y, (g^x)_P)$. Then he fetches the server's public key to encrypt his identity C, his MAC address MAC_C, a random nonce N_C, and the hash, and then he sends

$$(g^x)_P, H_{HK}(g^y, (g^x)_P),$$

$$E_S\{C, MAC_C, N_C, H_k(g^y, (g^x)_P)\} \tag{3.6}$$

to the server.

In this step, the client derives the session key k by Diffie-Hellman computation, which provides perfect forward secrecy for the protocol. The hash $H_{HK}(g^y, (g^x)_P)$ which serves as an authenticator as well as the two exponentials is sent back to the server so that the server does not need to store the two exponentials but still can verify that the two exponentials are not modified by checking the hash. Hence the server can resist DoS attacks that intend to deplete the server's storage space. The client's identity is protected with the server's public key to avoid identity disclosure.

4. After the server receives the above message, it verifies the validity of the hash $H_{HK}(g^y, (g^x)_P)$ by looking up the hash value in its storage. If the hash value is stored on the server, then the server can verify whether the two exponentials are valid by checking the received hash. After that, the server decrypts $E_S(C, MAC_C, N_C, H_k(g^y, (g^x)_P))$ and fetch the client's password according to his identity. The server now can decrypts $(g^x)_P$ to obtain g^x and computes the session key k in the same way as the client. At the end, the server assigns an IP

address IP_C to the client and sends the following message to the client.

$$e_k(C, S, IP_C, g^y, (g^x)_P) \tag{3.7}$$

In order for access control at the access point, the server also sends the session key k, the client's MAC address and IP address to the access point through a secure channel.

After successful authentication and key agreement, the enforcement of access control in our protocol uses an access control extension header. Specifically, we follow the approach of the Authentication Header in IPSec [27]. Our access control extension header is shown in Fig.3.5, where the Integrity Check Value (ICV) is a keyed hash function output given by

$$ICV = H_k(MACHeader||IPv6Header||$$

$$V||T||Res||Payload).$$

Here V denotes protocol version, T denotes the type of service, and Res denotes the reserved bits.

The ICV is computed by the client C for every IP packet it sends to the network, and this provides integrity and data origin authentication for the IP packet. It can also be used to provide protection against replays by incorporating a sequence number in the extension header [27].

When an IP packet is received from the wireless network, the access router looks

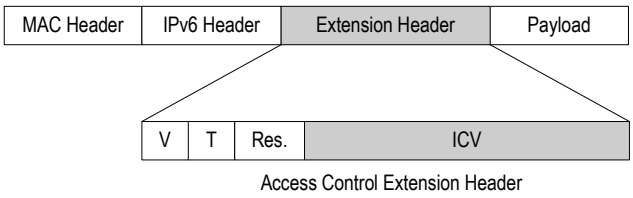


Figure 3.5: The Packet Header Format in Our Protocol.

up the MAC address in the ACL. If the entry exists for the client device, the access router fetches the secret session key, computes the ICV over the appropriate fields of the received packet, using the same formula as the client, and verifies that it is the same as the ICV included in the received packet. If the verification is successful, the access control extension header containing the ICV is stripped off and the packet is passed on; otherwise, the packet is dropped silently.

3.2.5 Security Analysis of Our Protocol

In this section, we analyze the security of our protocol and show that our protocol fulfills all the requirements aforementioned, including client identity confidentiality and resistance to DoS attacks.

Mutual authentication between the client and the server is achieved after successful protocol execution. The server authenticates the client by verifying the hash $H_k(g^y, (g^x)_P)$ encrypted with the server's public key in message (3.6), since only the legitimate client knows x and can compute the session key k. On the other hand, because only the valid server can decrypt the ciphertext in message (3.6) and know the client's identity, the client can authenticate the server by checking message (3.7). At the same time, both parties are ensured that the other party obtains the same session k as himself.

Our protocol provides perfect forward secrecy by employing Diffie-Hellman key exchange, and this ensures security of previous sessions even when the shared password or the server's private key is compromised. With the secret password P, an adversary who has stored previous communication content can decrypt $(g^x)_P$ from previous session. If the server's private key is compromised, the adversary can discover the client's identity. But in both cases, it is still computationally infeasible for the adversary to obtain k assuming the hardness of discrete logarithm. So the adversary still cannot derive the session key and in turn cannot disclose previous communication.

In our protocol, the identity of the client is protected against both passive and active attacks. After the client receives message (3.5), it fetches the public key of the server according to the server's identity, and sends its identity encrypted by the server's public key. Hence only the valid server who holds the corresponding private key can decrypt it to obtain the client's identity. Later in message (3.7), the client's identity is protected with the session key k. Since a passive attacker cannot complete Diffie-Hellman computation to derive k, he cannot obtain any information about the client's identity. On the other hand, an active attacker has no legitimate private key to decrypt the identity information in message (3.6), and he cannot complete Diffie-Hellman exchange to derive k to decrypt message (3.7) either. Therefore, both passive and active attacks cannot disclose the client's identity. However, it should note that the MAC address of the client's machine can be used to identify the client if he always uses the same machine. Only if the client is capable of changing his MAC address can he keep him totally anonymous to outsiders.

The shared password is secure against off-line dictionary attacks in our protocol. In

the protocol, the password is used to encrypt the random exponential g^x generated by the client, and hence an adversary cannot verify his guess because he does not know g^x. If the adversary impersonates as the server, and sends an exponential $g^{y'}$ to the client. The client then will derive the session key $k' = H(C|S|g^{xy'})$, which can be computed by the adversary who has y'. However, the adversary cannot verify his guess by checking $E_S\{C, MAC_C, N_C, H_k(g^y, (g^x)_P)\}$ because N_C is a random nonce chosen by the client.

The server does not have to keep any state and commit any storage when sending out message (3.5), and this relieves the server from memory DoS attacks that intend to exhaust the server's memory. The two exponentials will be sent back in message (3.6), so the server does not need to create state and store these information. Hence, if the client is fraudulent, the server will not have committed any storage resources. In order to avoid the case in which the exponentials may be modified, the server uses a secret hash key HK to compute an authenticator $H_{HK}(g^y, (g^x)_P)$. The key HK is private to the server and is updated frequently, and the authenticator sent back in message (3.6) can be used to ensure that the exponentials are the same as those in (3.5). On the other hand, the server is also protected from computation DoS attacks aiming to exhaust the server's computation resource. In message (3.5), the server's computation cost includes only a cryptographic hash operation and an exponential g^y. Note that the random exponential g^y can be computed beforehand or periodically computed when the server is lightly computational burdened. That is, the server only commits itself into expensive computation upon successful verification of $H_{HK}(g^y, (g^x)_P)$ after receiving message (3.6), and this effectively reduces threat of DoS attacks by delaying the server's computation to the last round. Moreover, when the server is under the computation

DoS attack and heavily burdened in computation, the server can reuse previously used exponential g^y. While if the adversary launches a computation DoS attack by flooding message (3.6) to the server, the server just resends message (3.7) to the other party.

Only the authorized client who has the valid password P can establish a secret key K with the authentication server and in turn with an access router. Since only the client with K can compute ICV for an IP packet, only the client can access the network services. Note that the ICV is computed over the MAC header and the entire IP packet; hence, any modification to the MAC header and the IP packet during transmission will be detected by the access router. This ensures that only authorized parties can gain access to the wireless network.

3.2.6 Implementation and Performance Analysis

While providing desirable features for wireless networks, our protocol also achieves great computation efficiency for wireless networks. In our protocol, the server only needs 2 exponentiations and 1 public key decryption, and the client requires to compute 2 exponentiations and 1 public key encryption. We evaluate the performance of our protocol by measuring the overhead of our protocol. The overhead incurred by our access control protocol consists of two parts. The first part of the overhead comes from authentication and key exchange of the protocol. Before a client can access the wireless network, it needs to follow the protocol with the authentication server to agree on a session key for each session. This part of overhead is associated with every session. Thereafter, the client uses the session key to encrypt and authenticate every packet, while the access router verifies every packet from the client with the same session key. The delay of the

Table 3.1: Benchmarks for Cryptographic Operations

	450MHz P III [1]	2.1GHz P IV
Modular Exp.	8.80ms	3.69ms
RSA Dec/Sig	17.00ms	4.63ms
RSA Enc/Ver	0.81ms	0.18ms
HMAC/MD5	480Mb/s	1726Mb/s
AES	243Mb/s	496Mb/s

[1]The benchmarks come from [5, 38, 40] and [43].

packet processing leads to the second part of the overhead for the wireless network, and it is associated with every packet.

We implement our protocol on the network simulator ns2 to evaluate the performance of our protocol. In our implementation, we adopt the benchmarks for the cryptographic operations on two different hardware platforms. One is a 450MHz Pentium III processor [5, 38, 40], and the other is a 2.1GHz Pentium IV processor [43]. The processing times of the required cryptographic operations, such as modular exponentiation, RSA decryption/signature and encryption/verification, are listed in table 3.1. These benchmarks are also used to evaluate our PKC-based protocol later. We assume the following system setup for performance evaluation of our protocol: AES is used for encryption and HMAC/MD5 is used for ICV calculation; the bandwidth of the wireless network is 1Mb/s; the random nonces N_C and N_S, the exponents, the identity of each party are 160-bit long; the modulus p is 1,024-bit long.

For the first part of the overhead, the time on hash computation, random number generation, modular multiplication and modular inversion can be ignored, since it is relatively much smaller than the time on Diffie-Hellman key-pair generation and key agreement.

After successful authentication and key exchange, the client obtains the session key to secure its subsequent communications. With the session key, every packet is encrypted and an integrity check value (ICV) of the packet is calculated for the purpose of authentication. After the packet is received by the access router, the router decrypts the packet and computes the ICV of the packet for authentication. These operations incur the second part of overhead for our protocol.

We simulate the protocol and obtain the overhead of sessions with 1000 1000-bytes packets. The total overhead of our protocol is calculated and listed in table 3.2. As seen from the table, the total overhead of our protocol takes up only 3.77% for the 450MHz Pentium III and 2.86% for the 2.1GHz Pentium IV of the total time. Therefore, our protocol can be used to secure wireless communications with degrading the performance slightly.

Table 3.2: Overhead of Our Password Based Protocol

	450MHz P III	2.1GHz P IV
Overhead/Session	53.0ms/14.6ms [1]	19.6ms/14.6ms
Overhead/Packet	0.099ms/0.195ms	0.050ms/0.195ms
Total Overhead [2]	$(67.6 + n \cdot 0.294)$ms	$(34.2 + n \cdot 0.245)$ms
Total Overhead [3]	3.77%	2.86%

[1] The data is in form of computation/transmission time.
[2] The total overhead of a n-packet session.
[3] The total overhead of a 1000-packet session as a percentage of the total time.

3.3 PKC-based Authentication and Key Exchange for Wireless Networks

In the last section, we focused on password based access control protocols. Such protocols are easy to use especially for mobile users who move from one device to another device. In this section, we are concerned with access control protocols based on public key cryptosystem (PKC). For this purpose, we first review the Stanford access control architecture and protocol as proposed by Faria and Cheriton [22]. We then point out some weaknesses in the Stanford protocol and present our access control protocol.

3.3.1 The Stanford Access Control Architecture

Faria and Cheriton [22] introduced a PKC-based two-layer architecture (see Fig. 3.6) for secure access to wireless 802.11 networks. The protocol stack of the architecture is composed of a Secure Internet Access Protocol (SIAP) and a Secure Link Access Protocol (SLAP). The SIAP, running at the application layer, provides mutual authentication and sets up fresh session keys between the SIAP client and SIAP server, while the SLAP, running on top of the data link layer, is responsible for data link access control using the session keys negotiated by SIAP. The SIAP/SLAP protocol stack resides at client devices and 802.11 access points. Therefore, there is no centralized authentication server as in the Lancaster architecture; the functionality of the authentication server is distributed among access points. This distributed nature makes PKC-based cryptographic solutions an ideal choice.

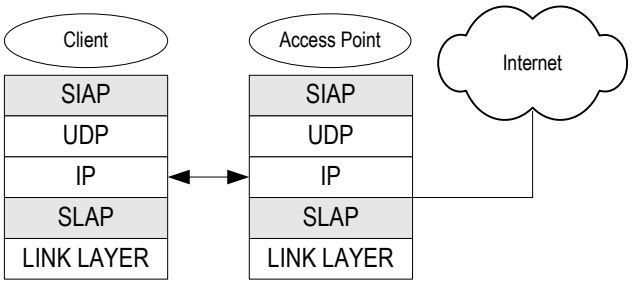


Figure 3.6: The Stanford Access Control Architecture.

3.3.2 Security Requirements

In addition to the requirements as given in Section 3.2.2, it is possible to design a PKC-based protocol to protect identity of the client and resist denial of service (DoS) attacks. That is,

- *Client Identity Confidentiality:* Confidentiality protection of a client's identity against both passive and active attacks;
- *Protection Against DoS Attacks:* The protocol should have certain built-in remedies to reduce the effect of DoS attacks aiming to exhaust the server's computation resource (computation-DoS) or storage resource (memory-DoS).

The requirement of identity confidentiality, or identity anonymity, originates from the movement of the mobile nodes. In the wireless environment, the current location and the movement of a roaming user are important parts of the user's privacy, and they should be protected during communications. Knowing the user's identity helps the attacker to locate the user and track his movement, so it is important for a protocol in a wireless environment to provide identity confidentiality to users.

With password based protocols it is difficult to protect a client's identity since the authentication server needs to know the client's identity to match against the client's password. The asymmetric nature of PKC, however, allows us to overcome this difficulty by encrypting the client's identity with the public key of the authentication server.

3.3.3 The SIAP/SLAP Protocol and Its Security Analysis

In the Stanford access control architecture, the Stanford protocol is intended to provide DoS protection but without the identity confidentiality for the wireless network. In their protocol, the SIAP protocol establishes secret session keys between a client and an access point, and then the session keys are passed to the SLAP, where the key is used for access control at the link layer.

To access the wireless network, a client C runs the SIAP with a SIAP server S (which resides at an access point) as follows.

1. The client C initiates the protocol by sending

$$ver, msg\_id \tag{3.8}$$

   to the SIAP server S, where ver is the protocol version and msg_id is the message identifier.

2. The server responds with a signed nonce N_S and its public key certificate $Cert_S$ signed by a well known certification authority,

$$S_S(ver, msg\_id, N_S), N_S, Cert_S. \tag{3.9}$$

3. After the client receives the above message, it first verifies the validity of the SIAP's public key using the public key of the certification authority and then verifies the validity of the signature using SIAP's public key. The client generates a nonce N_C and sends

$$S_C(ver, msg\_id, N_C, N_S, MAC_C), N_C, Cert_C \tag{3.10}$$

to the SIAP server. In this message, $Cert_C$ is the client's public key certificate and MAC_C stands for the client's MAC address which can be extracted from the header of the data link packet.

4. Upon reception and successful verification of the client's signature, the server sends

$$S_S(ver, msg\_id, N_C, IP_C, E_C(K), T),$$

$$IP_C, E_C(K), T \tag{3.11}$$

to the client. In this message, K is a secret value from which the client and the server generates session encryption key and integrity key, $E_C(K)$ is the encryption of K using the client's public key, IP_C is the IP address assigned to the client by the server, and T is a ticket which is used to propagate its state to other servers in the network. The function of the ticket T is not discussed since it is out of the scope of this Chapter.

5. The client verifies the server's signature in the above message, and if the result is positive, the SIAP execution is considered successful. At this point, the client

and the server each generates an encryption key and an integrity key based on K and passes the two keys from SIAP to SLAP for the purpose of access control at the data link layer.

The complete protocol is illustrated in Fig. 3.7.

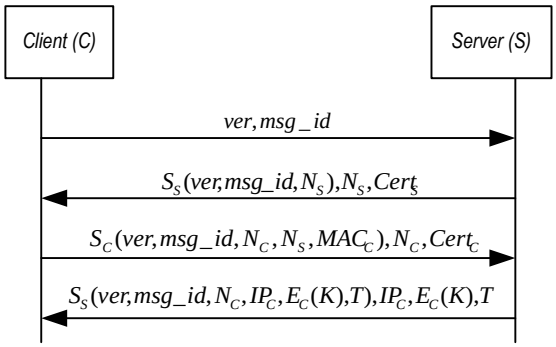


Figure 3.7: The SIAP Protocol.

After successful execution of SIAP between the client and the server, the generated keys, including an encryption key and an integrity key, are passed from SIAP to SLAP. As the client is sending out packets, the IP header and the payload of every packet are encrypted using AES with the encryption key, and then an integrity check value (ICV) is calculated over SLAP header, IP header and the payload using HMAC-MD5 with the integrity key. The ICV is copied into the MAC header of the packet. The packet after processed by SLAP is illustrated in Fig. 3.8.

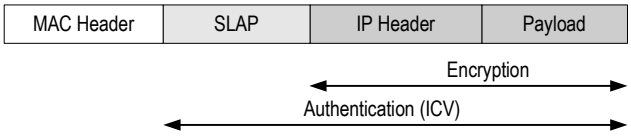


Figure 3.8: The SLAP Packet

For each packet received by the server, the server fetches the integrity key and

encryption key corresponding to the sender's MAC address. If the integrity check of the packet is successful, the server proceeds to decrypt the encrypted part, and pass the packet to the intranet. As a result, only authorized packets can pass the check and be delivered to the intranet.

The SIAP provides certain resistance to DoS attacks in the sense that SIAP servers are distributed among access points. However, individual SIAP server is defenseless against both computation-DoS and memory-DoS attacks. The computation-DoS attack can be carried out simply by sending a large number of message (3.8) to the SIAP server. Each message (3.8) will launch the server into computational expensive signature generation operations.

The SIAP does not possess perfect forward secrecy since the secret K is encrypted by the client's public key. Once the client's private key is compromised, all the past session keys are exposed and all past communication content is revealed. To offer perfect forward secrecy in the protocol, a Diffie-Hellman key exchange is needed as in our PKC-based protocol proposed later.

Lastly, in SIAP, the client's public key certificate is sent in clear, therefore it does not provide client identity anonymity. The identity information can be used by an attacker to locate a mobile node and track its movement.

3.3.4 Our Protocol for the Stanford Architecture

Aware of the design flaws of the Stanford protocol, we present a PKC-based authentication and key exchange protocol to replace it. The Internet draft, JFKi [1], has been proposed as an efficient, DoS-resistant key exchange protocol for Internet, and it also

provides active identity protection for the initiator (i.e., the client). The JFKi protocol employs a cookie mechanism in dealing with DoS attacks, and we borrow this mechanism to build a new protocol for access control in wireless networks. Compared with JFKi, the server in our protocol requires less computation before a round trip of messages are transmitted, and our protocol can protect the client's identity from disclosure even when secret exponents are disclosed. Hence, our protocol offers better resistance to DoS attacks and stronger identity protection.

As defined in Section III, let $\mathbb{G}_{p,q}$ be a large multiplicative group, over which the discrete logarithm problem is hard. Let g be a generator of $\mathbb{G}_{p,q}$. Our protocol is composed of four message exchanges as follows.

1. The client generates a nonce N_C, a random value $x \in_R \mathbb{Z}_q^*$, computes the exponential g^x and sends

$$N_C, g^x \tag{3.12}$$

   to the SIAP server. The exponential g^x is used for Diffie-Hellman exchange to provide perfect forward secrecy, while the nonce N_C allows the client to reuse the same exponential g^x in different sessions. In order to offer user identity confidentiality, no information related to the client's identity exists in this message. And this message is not authenticated and could be sent by an attacker who aims to mount a DoS attack.

2. Upon receipt of the above message, the server chooses its own nonce N_S, a random value $y \in_R \mathbb{Z}_q^*$ and the corresponding exponential g^y. The it calculates

an authenticator using a local secret HK and sends

$$N_C, N_S, g^y, S, H_{HK}(g^x||g^y||N_C||N_S||MAC_C) \tag{3.13}$$

to the client.

In this message, MAC_C is the client's MAC address, and HK is a secret hash key known only to the server and is updated frequently by the server. The $H_{HK}(g^y||N_C||N_S||MAC_C)$ functions as an authenticator and a cookie to be sent back by the client in the next message, and it enables the server not to store the two nonces and the two exponentials. When the authenticator, the two nonces and the two exponentials are sent back by the client in the message (3.14), the server checks the authenticator and is ensured that the two nonces and two exponentials are the same as in the message (3.13). The key pair (y, g^y) can be generated and stored by the server beforehand (or periodically), hence the server does not need to create any state at this stage and avoids the threat of memory-DoS attacks.

In order to resist computation-DoS, the server cannot perform computationally expensive operations since the server cannot determine whether the client is legitimate. By producing the exponential g^y periodically and reusing them when exhausted, the server needs minimal computation at this stage, including only a cryptographic hash function and the generation of a random nonce. As a result, computation-DoS attacks against our protocol are not effective at this point. Moreover, the server does not sign the exponential as in JFKi, and this

alleviate the server's computation effort in this step. Hence our protocol provides better resistance against DoS attacks than JFKi.

3. After receiving the above message, the client completes Diffie-Hellman computation to derive g^{xy}. Then the client computes a signature over the nonces N_C, N_S, the exponentials g^x, g^y, g^{xy} and the server's identity S. After that, the client encrypts this signature and the client's identity with the server's public key, and sends the ciphertext, the authenticator, the two nonces, and the two exponentials to the server as follows:

$$N_C, N_S, g^x, H_{HK}(g^x||g^y||N_C||N_S||MAC_C),$$

$$E_S(C, S_C(N_C, N_S, g^x, g^y, g^{xy}, S)). \tag{3.14}$$

   In this message, the two nonces and the two exponentials need to be sent back to the server because the server does not keep state for these information. The authenticator is used by the server to verify that the nonces and the exponentials are not substituted by an attacker. The identity and the signature of the client are encrypted with the server's public key so as to achieve identity anonymity for the client.

4. Upon receiving the above message, the server finds the stored key pair (y, g^y) with g^y, and verifies the two nonces and the two exponentials against the authenticator. If the verification is successful, the server decrypts the part encrypted with its public key to obtain the client's identity and the signature. Then the server completes the Diffie-Hellman computation to derive g^{xy}, and so it can authenti-

cate the client by the signature with the client's public key. After that, the server derives the session key as follows:

$$K = H(g^{xy}||N_C||N_S),$$

and sends following message,

$$e_K(IP_C, S_S(IP_C, N_C, N_S, g^x, g^y, g^{xy}, C)) \tag{3.15}$$

to the client, where IP_C is the IP address assigned to the client by the server. Before sending this message, the server is not authenticated by the client yet. Therefore, a signature of the server is sent in this message to authenticate the server to the client. In order to provide identity confidentiality for the client, the signature and the IP address IP_C are encrypted with the derived session key K. And that provides key confirmation for our protocol as well.

5. Upon receipt of this message, the client computes the session key $K = H(g^{xy}||N_C||N_S)$, and decrypts the message to obtain the IP address and the signature. Then the client verifies the signature, and the server is authenticated if the signature passes the verification. Finally, both the server and the client will pass the session key from the SIAP to the SLAP for access control at the data link layer.

After the server and client derive the same session key and pass the session key from the SIAP to the SLAP, every packet sent from the client to the server will be encrypted and authenticated with the session key. The packet format is the same as the one in

Fig. 3.8.

The complete protocol is illustrated in Fig. 3.9.

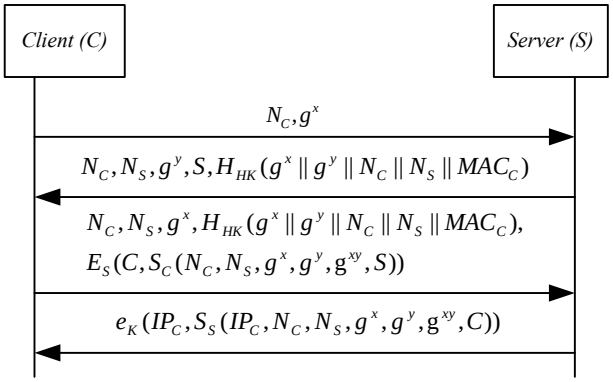


Figure 3.9: Our Protocol for the Stanford Architecture.

3.3.5 Security Analysis of Our Protocol

Our protocol is intended to provide mutual authentication, key authentication, key confirmation, key freshness, access control, perfect forward secrecy, client identity confidentiality and DoS resistance for wireless communications.

Mutual authentication is guaranteed by verifying the each other's signature. The client is authenticated to the server in the third message (3.14) by providing a signature $S_C(N_C, N_S, g^x, g^y, g^{xy}, S)$ to the server. On the other hand, the client can authenticate the server by verifying the encrypted signature $S_S(IP_C, N_C, N_S, g^x, g^y, g^{xy}, C)$ sent by the server in message (3.15). Both signatures are computed over two nonces and two random exponentials, so a reply attack is impossible to get an attacker authenticated successfully.

Our protocol also provides key authentication and key confirmation. The client can be ensured that the exponential g^y does indeed belong to a valid server by verifying the

signature in message (3.15). Also the server can verify the signature $S_C(N_C, N_S, g^x, g^y, g^{xy}, S)$ in message (3.14) to be ensured that the exponential g^x is indeed generated by a valid client. Implicit key confirmation is provided in the protocol in that both parties need to decrypt and verify the information protected by the session key K. This guarantees that both parties hold the same session key K.

Both the client and the server contribute to the session key K. As a result, the random nonces and exponentials chosen by the two parties ensure that the session key K is different every time, and hence guarantees key freshness for our protocol.

In the protocol, Diffie-Hellman exchange is employed to derive the session keys and hence provides perfect forward secrecy for our protocol. In the case that an adversary compromises the long-term secrets of both parties (i.e., the private keys of both parties), the adversary still cannot gain any useful information about past session keys. With the private keys of both parties, an adversary can decrypt the encrypted part in message (3.14), but this does not help the adversary to recover past session keys in any sense. Since the session keys are derived by Diffie-Hellman exchange between the client and the server in our protocol, it is infeasible for the adversary to recover past session keys under the assumption of hardness of discrete logarithm problem. Moreover, the amount of forward secrecy can be traded off against the resistance to DoS attacks [1]. When the server is under DoS attacks, it can reuse previous exponentials to mitigate the threat of the attacks by sacrificing some amount of perfect forward secrecy. Otherwise, the server can use different exponentials for each time to achieve perfect forward secrecy.

In our protocol, the identity of the client is protected against both passive and active attacks. After the client receives message (3.13), it fetches the public key of the server

according to the server's identity, and sends its identity encrypted by the server's public key. Hence only the valid server who holds the corresponding private key can decrypt it to obtain the client's identity. Later in message (3.15), the server's signature computed over the client's identity is also protected with the session key K. Since a passive attacker cannot complete Diffie-Hellman computation to derive K, he cannot obtain any information about the client's identity. On the other hand, an active attacker has no legitimate private key to decrypt the identity information in message (3.14), and he cannot complete Diffie-Hellman exchange to derive K to decrypt message (3.15) either. Therefore, both passive and active attacks cannot disclose the client's identity.

In the JFKi protocol, the client authenticates the server by verifying the server's signature over its exponential in the second message, and hence the client is ensured that the server's exponential is authentic. And then the client sends its identity protected by the keys derived from Diffie-Hellman exchange. This method used in JFKi to achieve identity confidentiality has two weaknesses. The first is that before a round trip occurs the server needs more computation including the generation of the exponential and a signature on its exponential. This makes the server more susceptible to computation-DoS attacks. The second is that once the attacker knows the secret exponent y of the server's exponential, the attacker can launch a replay attack to discover the client's identity. Compared with JFKi, the server in our protocol needs less computation in message (3.13), only including the generation of an exponential. For our protocol, even when the server's secret exponent y is known to an attacker, the client's identity confidentiality is still guaranteed as long as the server's private key is not exposed. In our protocol, we use the server's private key to protect the client's identity because the

server's private key is a long-term secret and more secure than the secret exponent y.

The server does not need to keep any state when sending out message (3.13), and this avoids the threat of memory-DoS attacks. The nonce N_S and the exponentials g^y generated by the server will be sent back in message (3.14), so there is no need for the server to create state and store these information. Therefore, if the client is fraudulent, the server will not have committed these storage resources, which provides memory-DoS resistance. In order to avoid that the client may return forged nonce or exponential, the server employs a secret hash key HK to compute an authenticator $H_{HK}(g^y||N_C||N_S||MAC_C)$. The key HK is private to the server and is updated frequently, and the authenticator sent back in message (3.14) can be used to ensure that the nonces and the exponentials are the same as those in (3.13). On the other hand, in message (3.13), the server's computation cost includes only a cryptographic hash operation, the generation of a random nonce N_S and an exponential g^y. Just like the password-based protocol we proposed earlier, the threat of DoS attacks is effectively reduced by delaying the server's expensive computation to the last round. When the server is under the computation-DoS attack and heavily burdened in computation, the server can reuse previously used exponential g^y. In this case, the server's computation cost will only include the hash operation and the generation of the random nonce N_S, and this enables the server to be resistant to computation-DoS attacks while sacrificing some amount of forward secrecy.

3.3.6 Implementation Issues and Performance Analysis

In implementation of our PKC-based protocol for wireless networks, the following issues should be noticed and dealt with cautions. Like our password based protocol, our PKC-based protocol has the same requirements on random number generation, security associations rekeying, retransmission mechanism. Besides, it is required that the private key HK of the authentication server be kept secret and updated frequently to avoid compromise. The key is used as the hash key to compute an authenticator ($H_{HK}(g^x||g^y||N_C||N_S||MAC_C)$) which functions as a cookie. Its compromise will lead to DoS attacks against the server. To offer resistance to DoS attacks, the random exponentials of the server could be pre-computed and reused if under heavy computational burden. If the server's exponentials are reused due to heavy computational burden, perfect forward secrecy is no longer provided but the server is relieved from computation. This trade-off between forward security and resistance to DoS attacks can be different for different applications, and it depends on requirements of different applications.

Compared to our previously proposed password based protocol, our PKC-based protocol requires signature computation and needs more computation. However, it provides features like nonrepudiation that the password-based one does not possess.

Likewise, the overhead of our PKC-based protocol also consists of two parts: the authentication and key exchange overhead for each session and the packet processing overhead for each packet. For each session, the computation cost of the client includes the generation of an exponential, a Diffie-Hellman computation, a signature, a public key encryption and a signature verification (other computation tasks are ignored since they

need much less time). On the other hand, the server's computation cost includes the generation of one exponential, a public key decryption, a Diffie-Hellman computation, a signature verification, and a signature.

With regard to the second part of overhead, the delay of packet processing is the same as that of the password based protocol discussed earlier. Here we also employ the same benchmarks and system parameters as used in Section III. The overhead of our PKC-based protocol is listed in table 3.3.

Table 3.3: Overhead of Our PKC Based Protocol

	450MHz P III	2.1GHz P IV
Overhead/Session	88.6ms/7.7ms [1]	29.2ms/7.7ms
Overhead/Packet	0.08ms/0.16ms	0.04ms/0.16ms
Total Overhead [2]	$(96.3+0.24 \cdot n)$ms	$(36.9+0.20 \cdot n)$ms
Total Overhead [3]	4.2%	3.0%

[1] The data is in form of computation/transmission time.
[2] The total overhead for n-packet transmission.
[3] The total overhead of a 1000-packet session as a percentage of the total time.

As can be seen from the above table, our PKC-based protocol has an overhead of 4.2% and 3.0% of the total time for the two platforms, respectively. As a result, it can be employed for access control in wireless networks without worsening the performance dramatically.

3.4 Summary

Security issues are crucial for wireless communications, and a secure and efficient access control mechanism is the first line of defense for secure wireless networking. In this Chapter we reviewed two two-layer access control architectures, the Lancaster architecture [23,39] and the Stanford architecture [22]. We analyzed the access control protocols

in the two architectures and identified various weaknesses and flaws.

Based on the same two-layer architectures, we proposed two access control protocols to replace them respectively. The first one is based on a weak password shared between a client and an authentication server, while the second one is based on public key cryptosystems. Both of them can provide mutual authentication, secure key exchange and perfect forward secrecy. Furthermore, the former provides protection against password dictionary attack; while the latter offers client identity confidentiality and resistance to DoS attacks.

CHAPTER 4

Group Key Agreement Protocol for Wireless Ad Hoc Networks

4.1 Introduction

Wireless ad hoc networks continue to enjoy a tremendous rise in popularity because of their convenience, flexibility and low deployment cost. As a form of group, security requirements for group communications in ad hoc networks, including confidentiality, data integrity, authentication and access control, have to be satisfied by group key management schemes. Though extensive research has been done on group key management due to the proliferation of group-oriented applications, existing solutions are not well-suited for ad hoc networks because of the special properties of such networks.

In wireless ad hoc networks, fixed infrastructures are usually unavailable and each node in the network can only communicate directly with other nodes within its power range. Hence some nodes are required to relay packets on behalf of a source node in order to deliver data to its destination. Moreover, nodes in ad hoc networks are usually resource-constrained in terms of power, bandwidth and computation capabilities. The unique properties of ad hoc networks impose stringent requirements on group key management schemes. The absence of infrastructure makes centralized group key management schemes inapplicable, while resource constraints on group members demand

highly efficient group key management schemes.

As stated in Chapter 2, group key management schemes can be classified into two categories: group key distribution schemes and group key agreement schemes. Since the entity responsible for key generation and distribution could likely become the single point of failure as well as the performance bottleneck, group key distribution schemes are not preferable in ad hoc group communications. Within group key agreement schemes, centralized schemes use a centralized server for group key management and are ruled out in ad hoc networks. Distributed group key agreement schemes do not require the existence of a centralized server and are the subject of our study in this Chapter.

Some group key management schemes (e. g., [75, 76]) use a tree based approach to manage keys as well as to reduce communication, computation and storage cost on maintaining keying and rekeying materials. Wong *et al.* [71] performed an extensive theoretical and experimental analysis on various types of key graphs and concluded that the most efficient key graph for group key management is a d-degree tree. However, those key management schemes employing a key tree hierarchy are inefficient if they do not exploit the network topology to further reduce communication cost. In this case, two nodes that are assigned to communicate heavy key management traffic may have more hops between them, while two nodes that are assigned to exchange less key management traffic may be close to each other. Moreover, multicast of key information is a frequent operation in group key management schemes, but the multicast operation would be very inefficient if the key tree is constructed regardless of the network topology and location of the group member nodes.

It has been noticed that designing a key tree structure which matches the network

topology can significantly reduce communication cost, and this is important for wireless networks which is error-prone and bandwidth-limited. Such a key tree hierarchy can localize the transmission of keying information and hence significantly reduces the communication cost of rekeying. Some solutions have been proposed for group key distribution in WLAN and ad hoc networks. However, these solutions are not well-suited for group key agreement in ad hoc networks because keying messages transmission pattern in group key agreement is different from that in group key distribution.

To improve communication efficiency in group key agreement schemes, we propose an efficient group key agreement scheme with a new key tree construction for ad hoc networks. We present a tree-transformation algorithm which constructs an efficient key tree from the underlying multicast tree of the ad hoc network. The algorithm transforms a multicast tree into a binary key tree which localizes keying information transmission in group key agreement. Our group key agreement scheme based on this key tree largely reduces the communication cost while keeping the computation cost at a very low level.

4.2 Our Group Key Agreement Scheme

In the dynamic groups such as ad hoc networks, membership is changing frequently. A group key agreement protocol for such dynamic groups needs to support the membership events as follows:

1. Join: a new member is added into the group
2. Leave: a member is removed from the group
3. Partition: the group is split into two or more subgroups

4. Merge: two or more subgroups is united as a single group

The key tree construction schemes of group key distribution can not be employed directly in group key agreement because of different message transmission pattern, while existing group key agreement schemes ignore the network topology in their key tree construction. Therefore, we attempt to construct the key tree which matches the network topology. Stem from this point, we propose our communication-efficient group key agreement scheme for ad hoc networks. Our group key agreement scheme unites the multicast tree and the key tree such that the network topology is closely related to the key tree structure. We achieve this integration by converting the multicast tree into a binary key tree. On our converted key tree, every subtree of the key tree corresponds to a multicast subgroup of the multicast tree. Since the key tree is constructed such that the subsequent multicast can be finished efficiently, our group key agreement scheme achieves great efficiency in communications. Moreover, our scheme has good computation efficiency for group partition which occurs frequently in unstable networks.

Before proceeding to the next section, some notations used in this Chapter are listed as follows:

n	number of group members
M_i	i-th group member; $i \in \{1, ..., n\}$
$\langle l, v \rangle$	the v-th node at the l-th level on a key tree
p	a large prime modulus
g	exponentiation base

4.2.1 The Key Tree Hierarchy

The key tree used in the tree-based group key agreement schemes is a binary tree on which each leaf represents a group member. Each interior node of the key tree has

exactly two children, and it is not associated with any group member. Actually, these interior nodes are logical and the whole key tree is called the logical key hierarchy. An example of the key tree used in TGDH is illustrated in the Fig. 4.1. The nodes are denoted $\langle l, v\rangle$, where $0 \leqslant v \leqslant 2^l - 1$ since each level l hosts at most 2^l nodes (the root is at the 0-th level). For convenience of presentation, the nodes are numbered as if they were on a perfectly balanced tree. Each node $\langle l, v\rangle$ on the key tree is associated with a key $K_{\langle l,v\rangle}$ and a corresponding blinded key $BK_{\langle l,v\rangle} = g^{K_{\langle l,v\rangle}} \mod p$. The key $K_{\langle l,v\rangle}$ of M_i at the leaf node $\langle l, v\rangle$ is chosen randomly by M_i, while the key of an interior node is derived from the keys of the interior node's two children by Diffie-Hellman computation. Specifically, the key of an interior node $\langle l, v\rangle$ is computed recursively as follows:

$$\begin{aligned} K_{\langle l,v\rangle} &= g^{K_{\langle l+1,2v\rangle} K_{\langle l+1,2v+1\rangle}} \mod p \\ &= (BK_{\langle l+1,2v\rangle})^{K_{\langle l+1,2v+1\rangle}} \mod p \\ &= (BK_{\langle l+1,2v+1\rangle})^{K_{\langle l+1,2v\rangle}} \mod p, \end{aligned}$$

where node $\langle l+1, 2v\rangle$ and node $\langle l+1, 2v+1\rangle$ are the two children of the node $\langle l, v\rangle$.

Therefore, computing a key at $\langle l, v\rangle$ requires the knowledge of the key of one child and the blinded key of the other child. The key $K_{\langle 0,0\rangle}$ at the root node is the group key known only to the group members.

To compute the group key, a member M_i needs to know a set of blinded keys, which form a set called the co-path CP_i. For example, the co-path of the member M_2 in Fig. 4.1 is $CP_2 = \{\langle 3, 0\rangle, \langle 2, 1\rangle, \langle 1, 1\rangle\}$. With the blinded keys in the co-path CP_i, the member M_i can compute a series of keys from itself to the root of the key tree, and

these keys form another set called key-path. For the same example, the member M_2's key-path is $KP_2 = \{\langle 3,1\rangle, \langle 2,0\rangle, \langle 1,0\rangle, \langle 0,0\rangle\}$.

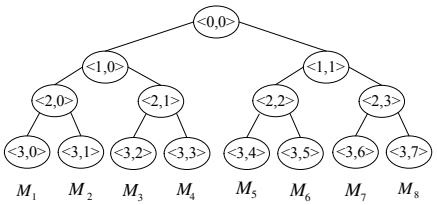


Figure 4.1: An Example of the Key Kree in TGDH

With all the blinded keys of its co-path, a member M_i can compute all the keys along the key-path, including the group secret $K_{\langle 0,0\rangle}$. For the example in Fig. 4.1, with its own key $K_{\langle 3,1\rangle}$, M_2 can compute $K_{\langle 2,0\rangle}$, $K_{\langle 1,0\rangle}$ and $K_{\langle 0,0\rangle}$ using $BK_{\langle 3,0\rangle}$, $BK_{\langle 2,1\rangle}$, $BK_{\langle 1,1\rangle}$, respectively.

In order for all group members to know the required blinded keys, multicast is frequently used for key information transmission in tree-based group key agreement. For instance, the blinded key $BK_{\langle 1,1\rangle}$ should be multicasted to M_1, M_2, M_3 and M_4, while the blinded key $BK_{\langle 1,0\rangle}$ needs to be multicasted to M_5, M_6, M_7 and M_8. However, during the construction of the key tree, the topology of the network is not taken into consideration, hence the key tree can lead to inefficiency in key information transmission.

4.2.2 The Multicast Tree Construction

A variety of tree-based multicast schemes [82–84] can be employed to achieve efficient multicast communication. In these multicast schemes, a multicast tree or a hierarchical structure is employed to achieve communication efficiency and scalability. In this phase, any efficient multicast tree construction scheme can be used for the group that wants to

generate a group key. If the ad hoc network has implemented the multicast tree algorithm for the purpose of multicast, our scheme can simply piggy-back on the multicast tree algorithm of the underlying multicast scheme.

Since any multicast tree algorithm can be employed in our scheme, we only illustrate our scheme with an example of a multicast tree based on the Steiner tree in [83]. The multicast tree algorithm generates a Steiner tree in an incremental way, and an example of the multicast tree based on the Steiner tree is illustrated in Fig. 4.2. The node M_1 is the group initiator and the other nodes $M_i (i = 2, ..., 8)$ are the rest members of the multicast group. For multicast, the source node M_1 multicasts the packets to its children M_2, M_3, M_4, and then the children further deliver the data packets to all nodes in their subgroups.

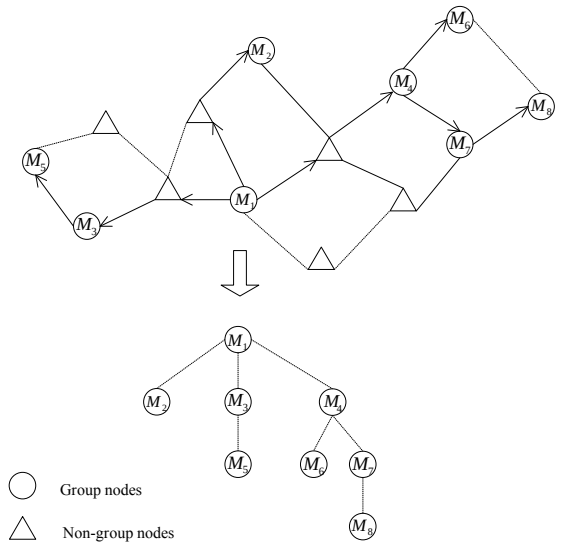


Figure 4.2: An Example of the Multicast Tree

On the multicast tree in Fig. 4.2, each edge is always connecting two nearest nodes in the ad hoc network, such as node M_3 and M_5 or M_7 and M_8. In other words, a node

is always near to its direct children(if it has), and it is also near to its direct parent(if it is not the root). While a node has a farther distance from its indirect children or parents. For instance, the distance between node M_1 and node M_7 is larger than that between node M_1 and M_4.

4.2.3 Conversion from the Multicast Tree to the Key Tree

Before the discussion of the conversion from the multicast tree to the key tree, we first analyze the traffic of the group key agreement.

We take the tree illustrated in Fig. 4.1 for example. We define the *sponsor* as the rightmost node of a subtree on the key tree. For example, node M_8 is the sponsor of the whole key tree, the subtree rooted at the node $\langle 1,1\rangle$ and the subtree rooted at the node $\langle 2,3\rangle$, while M_4 is the sponsor of the subtree rooted at the node $\langle 1,0\rangle$. These sponsors are responsible for multicasting blinded keys to the nodes that need them. For instance, M_8 should multicast $BK_{\langle 1,1\rangle}$ to M_1, M_2, M_3 and M_4, multicast $BK_{\langle 2,3\rangle}$ to M_5 and M_6.

As stated previously, each member M_i of the group needs to know the blinded keys of its co-path CP_i and its own session key. Consider the nodes on the subtree rooted at $\langle 2,0\rangle$ in our example. M_1 needs to know its co-path CP_1=$\{BK_{\langle 3,1\rangle}, BK_{\langle 2,1\rangle}, BK_{\langle 1,1\rangle}\}$; M_2 needs to know its co-path CP_2=$\{BK_{\langle 3,0\rangle}, BK_{\langle 2,1\rangle}, BK_{\langle 1,1\rangle}\}$ and $K_{\langle 3,1\rangle}$; and M_3 needs to know its co-path CP_3=$\{BK_{\langle 2,0\rangle}, BK_{\langle 1,1\rangle}\}$ and $K_{\langle 2,1\rangle}$. In other words, $BK_{\langle 1,1\rangle}$ needs to be multicast to M_1, M_2, M_3 and M_4; $BK_{\langle 2,1\rangle}$ needs to be multicast to M_1, M_2; $BK_{\langle 2,0\rangle}$ needs to be multicast to M_3 and M_4, so on and so forth.

Then the communication cost is listed in the following table:

It can be noticed that keying message transmission has a pattern of localization

Keys	Destination	Source Sponsor
$BK_{\langle 1,0\rangle}$	M_5, M_6, M_7, M_8	M_4
$BK_{\langle 1,1\rangle}$	M_1, M_2, M_3, M_4	M_8
$BK_{\langle 2,0\rangle}$	M_3, M_4	M_2
$BK_{\langle 2,1\rangle}$	M_1, M_2	M_4
$BK_{\langle 2,2\rangle}$	M_7, M_8	M_6
$BK_{\langle 2,3\rangle}$	M_5, M_6	M_8

since the keying messages are always useful to some nodes located on the same subtree. For instance, $BK_{\langle 1,1\rangle}$ sent by M_8 is only useful to M_1, M_2, M_3, M_4, which locate on the subtree rooted at $\langle 1, 0\rangle$; $BK_{\langle 2,1\rangle}$ sent by M_4 is only useful to M_1, M_2, which locate on the subtree rooted at $\langle 2, 0\rangle$. To efficiently deliver the information as listed in the above table, M_8 should be geographically close to nodes M_1, M_2, M_3, M_4, M_4 should be close to M_1 or M_2, and M_1 should be close to M_2, etc. After further consideration, it can be seen that efficient delivery requires M_8 be close to M_4, M_4 be close to M_2 and M_2 be close to M_1. Therefore, it can be deduced that two sponsors of two subtrees whose roots are siblings on the key tree should be geographically close to each other. For instance, M_4 and M_8, M_4 and M_2, M_6 and M_8 should be close to each other in Fig 4.1. This property of a key tree ensures a communication-efficient group key agreement scheme.

Since the multicast tree represents the network topology, we convert the multicast tree into the key tree to achieve our objective. We employ a conversion algorithm to convert the multicast tree in Fig. 4.2 into an efficient key tree.

The converted key tree has some desirable properties for efficient multicast. Each subtree of the multicast tree corresponds to a subtree on the converted key tree. Moreover, the root of the subtree on the multicast tree becomes the sponsor for the corre-

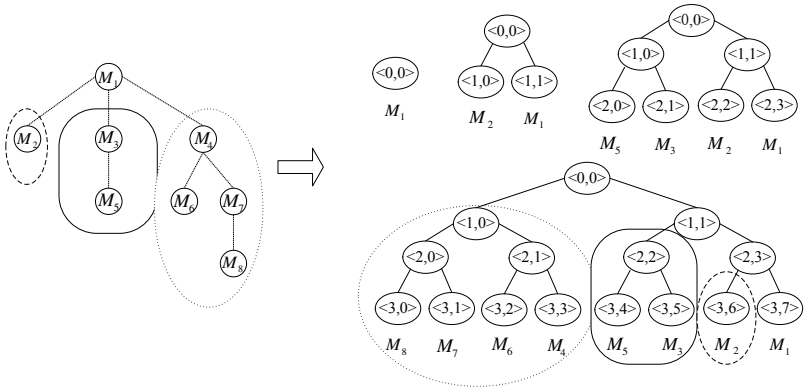


Figure 4.3: Conversion from the Multicast Tree to the Key Tree

sponding subtree on the key tree after conversion. For instance, the subtree rooted at node M_4 on the multicast tree is converted into the subtree rooted at node $\langle 1,0 \rangle$ on the key tree. Accordingly, the node M_4, which is a root on a subtree of the multicast tree, becomes the sponsor of the corresponding subtree on the key tree. Since all the nodes on the same subtree of the multicast tree are in close vicinity, a sponsor of the subtree can efficiently multicast information to the nodes on this subtree. For example, M_4, the sponsor of the subtree rooted at node $\langle 1,0 \rangle$, is close to all the nodes (M_6, M_7, M_8) on this subtree and hence can efficiently deliver multicast information to these nodes.

The second property of the converted key tree is that a sponsor is the parent of its sub-sponsor on the corresponding mutlicast tree. On the key tree, we define a sponsor A as a sub-sponsor of another sponsor B only if the root of A's corresponding subtree is a child of the root of B's corresponding subtree. In our example, node M_1 on the key tree has the sub-sponsors M_2, M_3, M_4, which are the children of M_1 on the multicast tree. This property ensures the efficient transmission of the group key agreement messages. For instance, the blinded key $BK_{\langle 1,1 \rangle}$ should be transmitted from M_1 to M_4, M_6, M_7, M_8,

so this can be accomplished by sending the blinded key from M_1 to M_4 and in turn sending it from M_4 to M_6, M_7, M_8.

Another advantage of the key tree is that every pair of leaf nodes that share the same parent take on a parent-child relationship on the multicast tree. Specifically, on the multicast tree, the right leaf node is the parent while the left node is the child. This advantage enables the two nodes to accomplish the Diffie-Hellman exchange efficiently since they are neighbors on the multicast tree.

Our scheme has another advantage on merging and partitioning over other tree-based Diffie-Hellman protocols. The group merge and partition occurs often in an unstable network like ad hoc networks. In our scheme, the computation cost can be greatly decreased because our key tree structure is based on the network topology as discussed later.

After the key tree is constructed as specified above, the sponsors on the key tree are responsible for multicasting required key information to each node. Then each node computes the group key as discussed earlier. The process of group key computation is the same as that of TGDH, and hence our protocol obtains the same security properties as TGDH.

Key Tree Balance Optimization

In the process of group key agreement, the computation cost is determined by the key tree height when the number of leaf nodes is fixed. Since our key tree is not constructed as a balanced binary tree, its height could be much higher than that of TGDH's key tree because the latter one is a balanced tree. As a result, our key tree construction can

lead to more complex computation than TGDH. Therefore, we propose the following optimization technique to reduce our key tree height and computation cost in group key agreement. The balance improved key tree has approximately the same height as a complete balanced tree in TGDH.

Suppose there are n nodes in the ad hoc network, and a multicast tree consisting of these n nodes has been constructed. Then we find an edge e_1 which can split the multicast tree into two subtrees S_1 and S_2 so that $| \, |S_1| - |S_2| \, |$ is minimized. $|S|$ denotes the number of nodes within S. Then the nodes from S_1 and S_2 would be assigned on the left and the right part of the key tree, respectively. For each subtree, similarly, we then find another edge e_2 which divides the subtree into two smaller subtrees with approximately equal number of nodes. And the two smaller subtrees also corresponds to two subtrees on the key tree. We continue the process until each subtree size is 1. Finally each node's position on the key tree is determined from the described process. An example of the optimization process is illustrated in Fig. 4.4.

This optimization process can largely reduce the key tree height, and the final tree height could be only $log(n)$ in some instances. The key tree height can not always be reduced to the optimal in order to preserve the advantageous features of our key tree. Basically, this is a trade-off between communication cost and computation cost.

4.2.4 Join and Leave Operations

Consider the scenario that a new node M_{n+1} wants to join a group $\{M_1, ..., M_n\}$. To preserve the advantages of our scheme, the new node should be added to the key tree in a special way. In the TGDH protocol, the node is added into the key tree such that

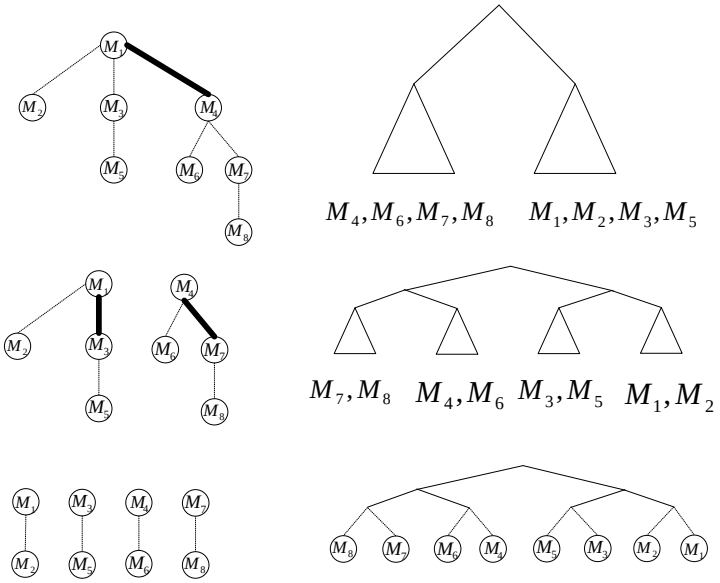


Figure 4.4: Key Tree Balance Optimization

the join does not increase the height of the key tree. This join approach would deprive the advantages of our key tree and break its special structure. In order to keep the advantages of our key tree structure, the new node is added into the key tree according to its location in the network.

The new node M_{n+1} starts the join process with a neighbor discovery procedure. With the neighbor discovery procedure, M_{n+1} recognizes its relative location in the group. Based on this knowledge, M_{n+1} identifies the insertion node on the key tree as well as on the multicast tree. Then M_{n+1} is inserted into the key tree and a sponsor informs other nodes about M_{n+1} joining. After the insertion, it seems as if the new key tree after joining of M_{n+1} is constructed directly from the $n+1$ nodes.

The join of M_{n+1} will result in two different scenarios according to the location of M_{n+1}. In the first scenario, the node M_{n+1} is located in the middle of two physically

neighboring group nodes. As can be seen from the construction of the key tree, there is a parent-child relationship between these two neighboring nodes on the multicast tree. On the corresponding key tree, the child node is a sub-sponsor of the parent node. In this case, the node M_{n+1} becomes the child of the parent node and the parent of the child node on the new multicast tree after its insertion. While on the corresponding new key tree, M_{n+1} becomes the sub-sponsor of the original sponsor and the original sub-sponsor becomes M_{n+1}'s sub-sponsor. As illustrated in Fig. 4.5, if the new node M_9 is located between M_1 and M_4, then M_9 is inserted into the multicast tree such that M_9 is the child of M_1 and the parent of M_4. While on the new key tree, M_9 becomes M_1's sub-sponsor and M_4 becomes M_9's sub-sponsor.

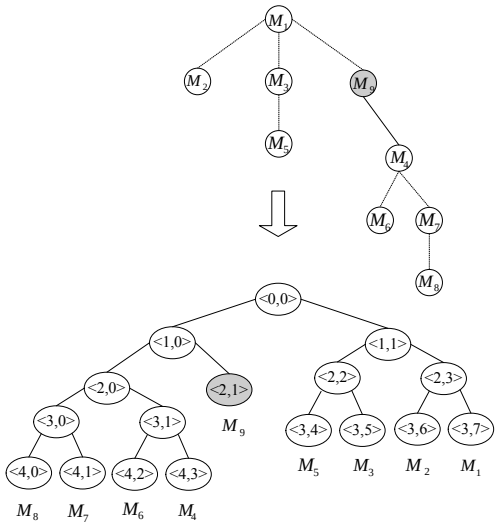


Figure 4.5: Join Operations: Scenario 1

In the second scenario, M_{n+1} is not located in the middle of any two neighboring group nodes and the insertion would increase the number of branches of the closest sponsor. The new node finds the closest sponsor by neighbor discovery procedure, and

then it is inserted into the subtree of the sponsor. The insertion position of the new node is a trade-off between computation and tree balance. As shown in Fig. 4.6, the join process is similar to the previous one.

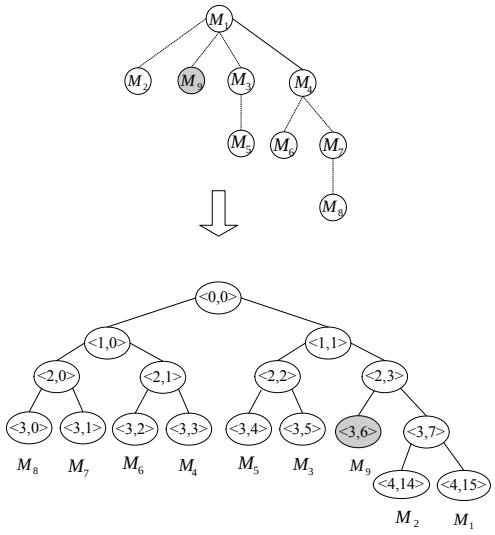


Figure 4.6: Join Operations: Scenario 2

After the new node M_{n+1} is inserted into the key tree, some key information needs to be exchanged within the group so that every node can compute the new group key. Since our new key tree still has the advantages of communication efficiency, this process can be accomplished efficiently.

With regard to the leave process, the situation is a little more complex than the join process. The most complicated scenario is the one when the root node of the mutlicast tree (M_1 in our example) leaves the group. In this case, the multicast tree is split into several subtrees. Our approach requires the root of the first subtree(M_2 in our example) set up connections with other roots of the subtrees. Then a new multicast tree and the corresponding key tree are constructed in the same way as illustrated in Fig. 4.7. In this

case, the key $K_{\langle 2,3\rangle}$ is changed to M_2's key and $K_{\langle 1,1\rangle}, K_{\langle 0,0\rangle}$ needs to be recomputed.

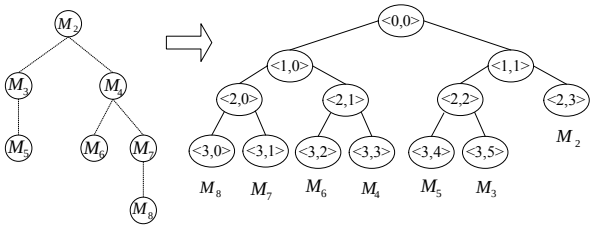


Figure 4.7: Leave Operations

For the leave event that the leaving node is not the root of the multicast tree, the leave process is much simpler. The leaving node is removed from the multicast tree, and the parent of the leaving node becomes the parent of the children of the leaving node. Then a new key tree can be obtained from the new multicast tree, and some computation and communication are required to derive the new group key.

After the above join or leave operations, the key tree has been updated and the group key has to be recomputed. The subsequent process is the same as the TGDH protocol, and hence our scheme obtains the same security properties as TGDH.

4.2.5 Partition and Merge Operations

In an unstable network where network failure occurs now and then, the whole group would be partitioned into several subgroups. Then after the network recovers from failure some time later, these disconnected subgroups can be reunited as a larger group. Since our key tree is constructed from the multicast tree which represents the network topology, our scheme exhibits extraordinary advantages over other tree-based Diffie-Hellman schemes.

When a network disconnection occurs in the network, the multicast tree is split

into several parts and the corresponding key tree is also split into parts accordingly. Some parts of the key tree are subtrees of the key tree, while others may be parts of subtrees. Within each partitioned subtree of the key tree, the key of the root of the subtree can be used as the group key within the corresponding group nodes. For those parts that are not subtrees, some computation is still needed to derive the new group key. The scenario that one link M_1 to M_4 is disconnected is illustrated in Fig 4.8. This disconnection partitions the network into two parts, and the corresponding key tree is split into two subtrees. As can be seen from this figure, nodes from either parts do not need any computation. The nodes on the subtree rooted at $\langle 1, 0\rangle$ can use the key $K_{\langle 1,0\rangle}$ as their new group key, while the other nodes can use the key $K_{\langle 1,1\rangle}$ as their group key.

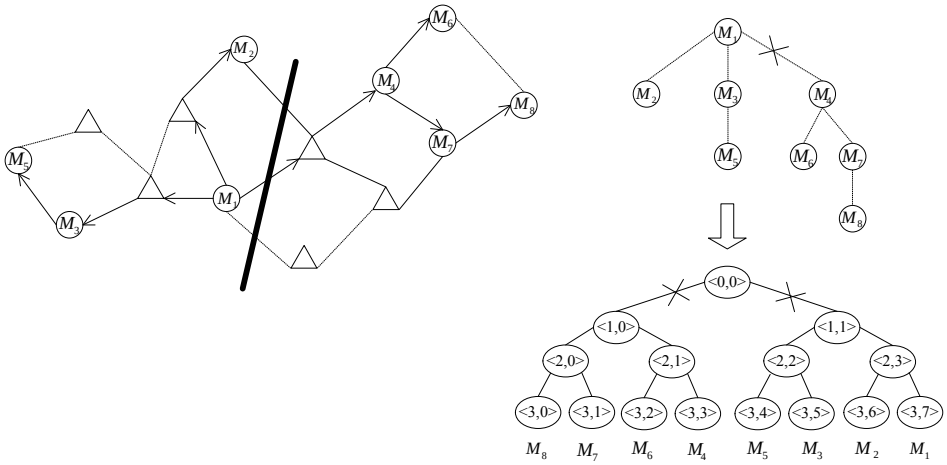


Figure 4.8: Partition of Key Tree in Our Scheme

While if the key tree is not constructed as our key tree(e.g. the key trees of STR [76] and TGDH [75]), the number of parts split by the disconnected links would be much more than that of our scheme. Therefore, considerable computation efforts are needed under this situation. An extreme situation is shown in Fig 4.9 where the key tree is completely

split by only one link failure. In this case, M_1, M_2, M_3, M_5 that can communicate each other in a subgroup need to reconstruct a small key tree for them, and then compute their new group key. M_4, M_6, M_7 and M_8 need to do the same operations.

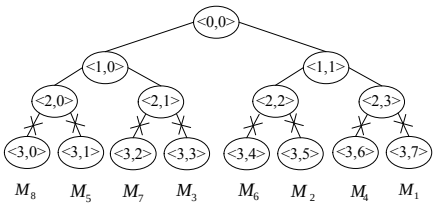


Figure 4.9: Partition of Key Tree in Other Schemes

Another scenario when the link from M_1 to M_3 is broken is shown in Fig 4.10. In this case, the key tree is split into two parts: the subtree rooted at $\langle 2, 2\rangle$ and the remaining part of the key tree. For nodes on the former part of the key tree, no computation is needed at all. While each node on the latter part still needs one exponentiation to generate a new group key. In this scenario, it is clear that our key tree requires less computation and communication cost than the key tree in Fig 4.9.

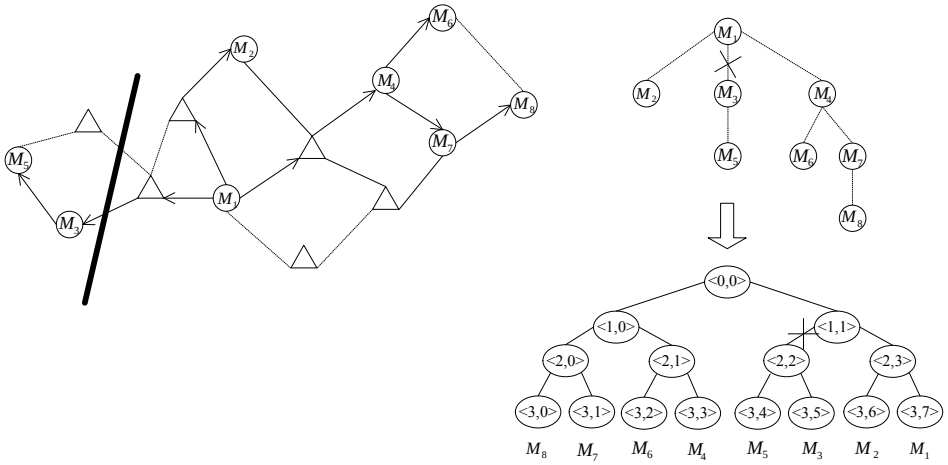


Figure 4.10: Another Partition Scenario

The merge process is the reverse of the partition process. Similar to the leave process, one of the roots of the subtrees on the multicast tree (i.e. the sponsors of the subtrees on the key tree) is responsible for establishing connections with other roots when merging. It is easy to see that the key tree after merging still possesses those advantages as stated earlier.

4.3 Discussion

4.3.1 Computation Complexity

We measure the computation complexity by the number of exponentiations in the tree-based group Diffie-Hellman key agreement schemes since exponentiation is the most computationally intensive operation. We compare our scheme with TGDH to evaluate the computation complexity of our scheme.

In the key tree construction phase, the computation complexity is close related to the key tree structure. The more balanced the key tree is, the more efficient in computation the group key agreement scheme is. The key tree of TGDH is constructed as a balanced binary tree, and the height of the key tree is minimized. While our key tree may be imbalanced since it is constructed based on the network topology. So in the construction phase, the computation cost of our scheme is more than that of TGDH, but this can be compensated with efficiency in communications of our scheme.

For the join event, the required computation cost is determined by the insertion position and the height of the key tree. Once a new node is inserted on the key tree, the keys on the key-path of the joining node need to be recomputed. On average, TGDH

needs $O(n)$ exponentiations in total for a join event. Since our key tree is not constructed to maximize the key tree balance, our protocol requires more computation than TGDH for node joining and leaving. Despite that, our simulation results presented later show that the total delay of computation and keying message transmission is much less than that of TGDH.

In our scheme, the computation cost is lowered as much as possible when the group is split into several group because of network failure. A network fault, such as a link failure, always partitions the network into several regions geographically. With our key tree construction approach, the nodes belonging to the same geographical region locate on the same subtree. Therefore, the keys of the roots of the subtrees still can be used instead of recomputing after partition, and computation cost is reduced dramatically. As illustrated in Fig. 4.8, even no computation is required under some circumstance, while TGDH needs much more computation in the same scenario as shown in Fig. 4.9.

When the network recovers from network fault, several groups need to be merged into a single group. Our scheme can reunite these groups into a larger group with the least computational effort while still enjoying all the advantages stated previously.

4.3.2 Communication Complexity

Efficiency in communication is the main advantage of our scheme and our main design objective. Since we have illustrated the advantage of our scheme for partition and merge, now we only discuss our scheme in the case of key tree construction, join and leave.

During the key tree construction phase, our key tree implements efficient delivery of keying messages. On the key tree, a sponsor of any group can always efficiently

multicast keying messages to all the group members. As shown in Fig. 4.3, the nodes on the subtree rooted at the node $\langle 1,0\rangle$ have a common node $\langle 1,1\rangle$ in their co-paths. So the blinded key $BK_{\langle 1,1\rangle}$ of the node $\langle 1,1\rangle$ should be multicast to these nodes(M_4, M_6, M_7 and M_8). The blinded key $BK_{\langle 1,1\rangle}$ is first unicast to M_4 by M_1, and is subsequently multicast to M_6, M_7 by M_4. M_7 then further unicast the blinded key to M_8. Moreover, the nodes M_4 and M_6, as well as the nodes M_7 and M_8, are close such that the keying message exchange between them is efficient.

If the key tree of TGDH is constructed as the one in Fig. 4.9, the communication cost would be significantly more than that of our scheme. In this case the nodes on the subtree rooted at $\langle 1,0\rangle$ are M_3, M_7, M_5 and M_8. Furthermore, M_3 and M_7 need to exchange blinded keys between them, and so it is with M_5 and M_8. However, the nodes M_3, M_5 are far from M_7, M_8, which makes the information transmission very inefficient.

We use the amount of keying messages transmission (unicast or multicast) during the protocol execution as the metric for communication performance evaluation. The amount of keying message transmission is determined by both the number of keys transmitted and the hops these keys traverse. Since all the key have the same size, we define one key being transmitted over one hop as 1 unit of communication cost. For a group of size $n = 2^L$, the communication cost for constructing a group key can be calculated as

$$Cost_C = \sum_{i=1}^{L} (2^i * MCost_{2^{L-i}}), \tag{4.1}$$

where $MCost_{2^i}$ is the cost of keying message multicast. The multicast cost $MCost_{2^i}$ is proportional to the multicast size $(2^i)^k$ [91], where k is a multicast cost parameter with

$0 < k < 1$.

The communication cost for group key construction in TGDH can be calculated as

$$Cost'_C = \sum_{i=1}^{L} (2^i * MCost_{2^L}), \tag{4.2}$$

$$\begin{aligned} Cost'_C/Cost_C &= \sum_{i=1}^{L} (2^i * MCost_{2^L}) / \sum_{i=1}^{L} (2^i * MCost_{2^{L-i}}) \\ &= \frac{2^{(1-k)} - 1}{2^{L(1-k)} - 1} \cdot 2^k \cdot (2^L - 1) \end{aligned}$$

For multicast cost parameter $k = 0.5$ and $L = 8$, the cost of TGDH $Cost'_C$ is about 10 times of that of our scheme.

The join and leave events are managed efficiently under our scheme. Take Fig. 4.5 as an example. After the new node M_9 is inserted into the key tree, M_9's blinded key $BK_{\langle 2,1\rangle}$ needs to be multicast to the nodes on the subtree rooted at $\langle 2, 0\rangle$. Then $K_{\langle 1,0\rangle}$ and $BK_{\langle 1,0\rangle}$ are recomputed, and $BK_{\langle 1,0\rangle}$ needs to be multicast to the nodes on the subtree rooted at $\langle 1, 1\rangle$. All these can be accomplished efficiently in the same way as in the key tree construction phase discussed above. Furthermore, after the insertion of the new node, the new key tree still obtains all the advantages as before. The leave operations can also be completed in an efficient way accordingly.

For a group of size $n = 2^L$, we can also calculate our scheme's communication cost

for a node's joining or leaving the group as follows:

$$Cost_{J|L} = \sum_{i=1}^{L-1} M_{2^i} = arg * \sum_{i=1}^{L-1} (2^{i \cdot k}), \tag{4.3}$$

while the cost for TGDH's communication cost for a node's joining or leaving is as follows:

$$Cost'_{J|L} = \sum_{i=1}^{L-1} M_{2^L} = arg * \sum_{i=1}^{L-1} (2^{L \cdot k}), \tag{4.4}$$

where arg is a fixed coefficient and k is the cost parameter of multicast. So it is clear that TGDH's communication cost for joining and leaving is much more than that of our scheme. $Cost_{J|L} : Cost'_{J|L} = 1 : (L-1)$ if $k = 1$, which means TGDH requires 5 times communication cost as our scheme if $n = 64$.

4.4 Implementation and Performance Evaluation

To evaluate the performance of our protocol, we have implemented our group key agreement protocol and the TGDH protocol on ns-2 for comparison. Our implementation is based on the MAODV protocol [85], which is a tree-based multicast protocol using AODV routing protocol [63] for ad hoc networks. Our group key agreement protocol uses the multicast tree of MAODV to construct the key tree.

We evaluate the performance of our group key agreement protocol in terms of group key agreement delay, communications cost and messages loss. The group key agreement delay measures the latency of each node to compute the group key after a node joins or leaves the group. We measure the delay as the period from the time a node requires

to join the group to the time a group member finishes the group key computation. The delay comprises of the time on group key computation and the communications latency. In our simulations, we assume that each node has the computation power of a 450MHz Pentium III processor, and a 1024-bit modular exponentiation cost 10ms for each node [86].

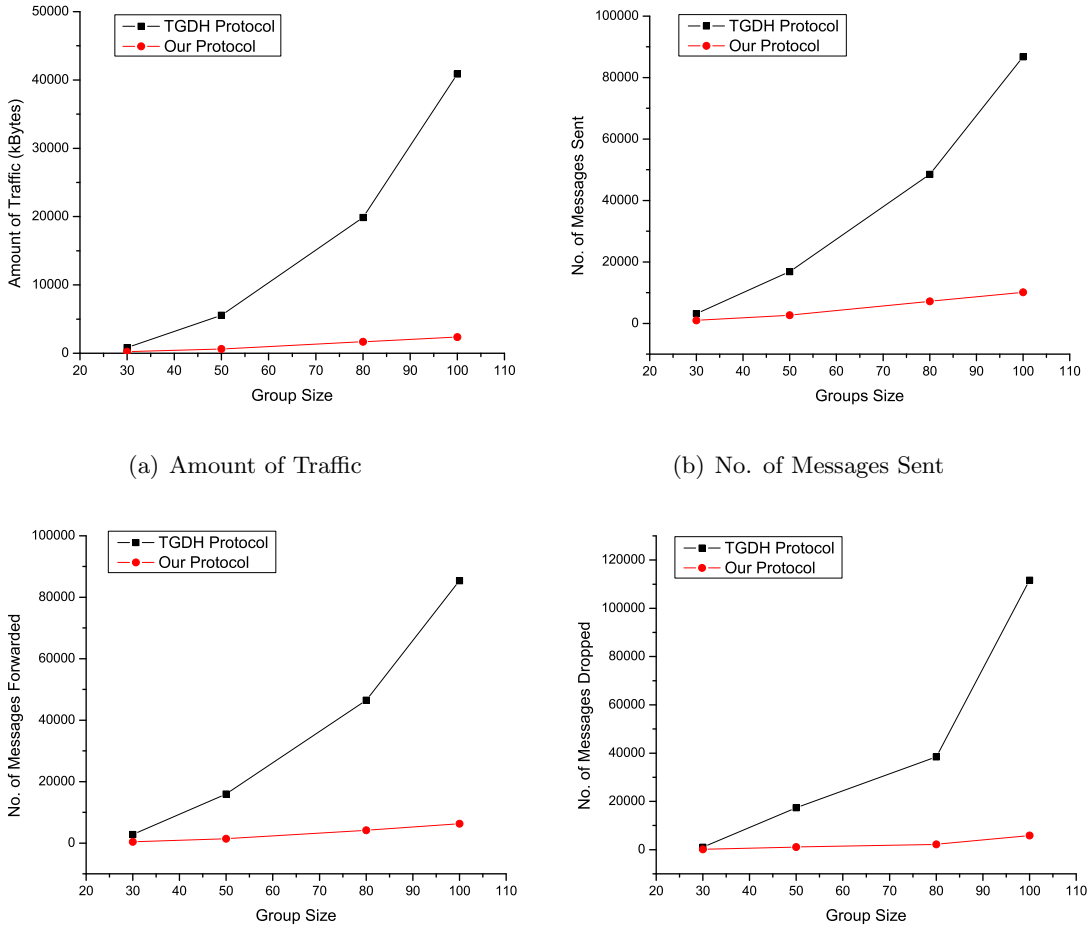
We simulate our protocol and the TGDH protocol on ad hoc networks with different network size ranging from 30 nodes to 100 nodes. The network size is the number of the nodes in the network, and it determines the maximum group size (when all the nodes join the group). When not all the network nodes join the group, the non-member nodes may need to relay packets for the group members to exchange key information. For each network size, we simulate both protocols on 20 different network scenarios. Then we analyze the join delay and the leave delay for different group sizes and different network sizes. The network size, connectivity and area of the network scenarios are listed in the following table.

Table 4.1: Connectivity of the Network Scenarios

Network Size	Average hops between two nodes	Maximum hops between two nodes	Simulation Area
30	3.557	12	1000mx1000m
50	3.227	9	1000mx1000m
80	5.547	18	1500mx1500m
100	4.819	15	1500mx1500m

For each experiment, we let the nodes join the group gradually until all the nodes join the group, and then let the nodes leave the group one by one until no one is in the group. Then we obtain the communication and group key agreement delay for the join

and leave events.



(a) Amount of Traffic

(b) No. of Messages Sent

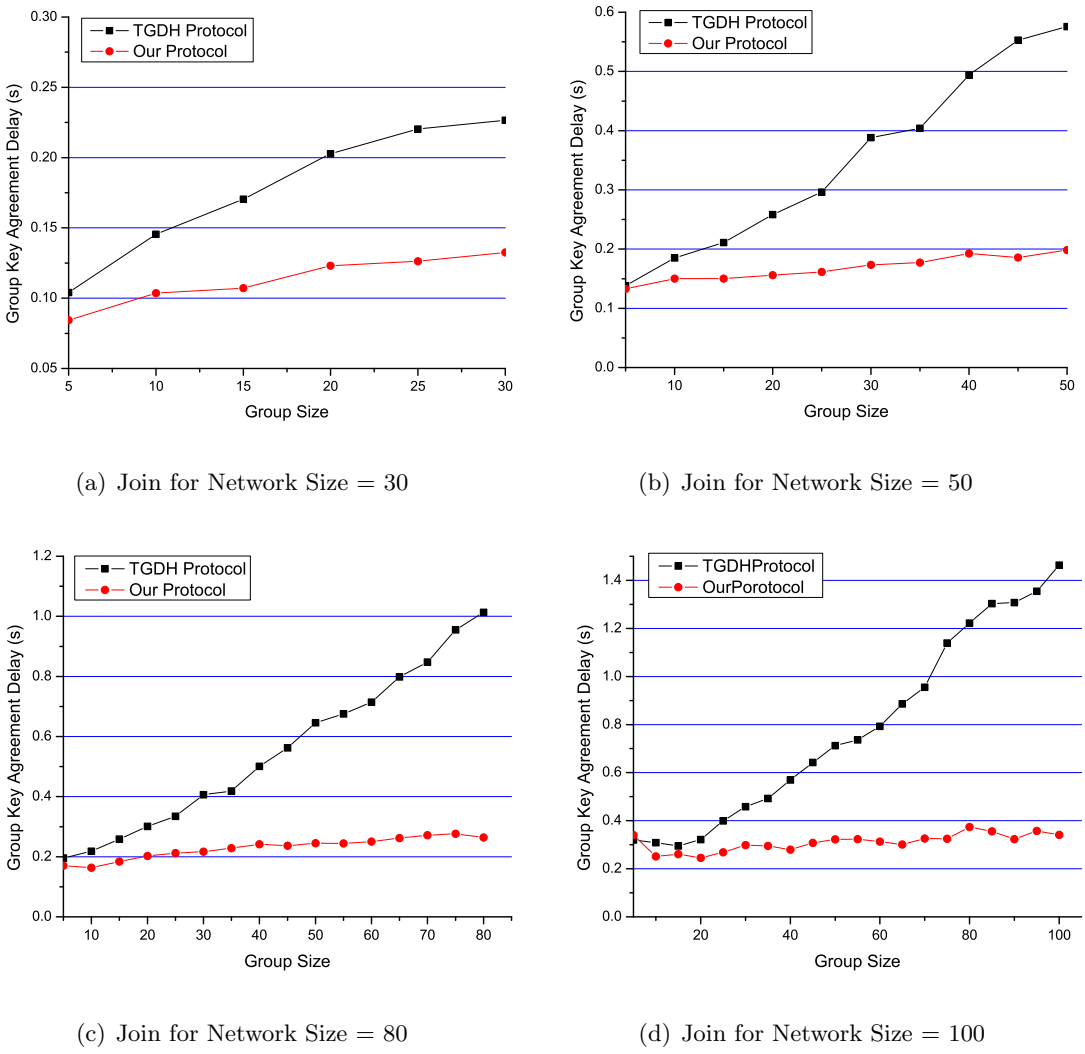
(c) No. of Messages Forwarded

(d) No. of Messages Dropped

Figure 4.11: Traffic Comparison Between TGDH and Our Protocol

Figure 4.11 shows the traffic comparison between TGDH and our protocol for different group sizes. Figure 4.11(a) illustrates the averaged total amount of traffic transmitted in the network on all the experiments for different network sizes, while figure 4.11(b), 4.11(c) and 4.11(d) show the averaged total messages sent, number of messages forwarded and number of messages dropped on all experiments, respectively. For the same network size, our protocol outperforms TGDH greatly on all the issues. Our pro-

tocol introduces a communication cost as much as 1/10 of TGDH as shown in the figure. And proportionally fewer messages and less messages are forwarded or dropped in the network with our protocol.



(a) Join for Network Size = 30

(b) Join for Network Size = 50

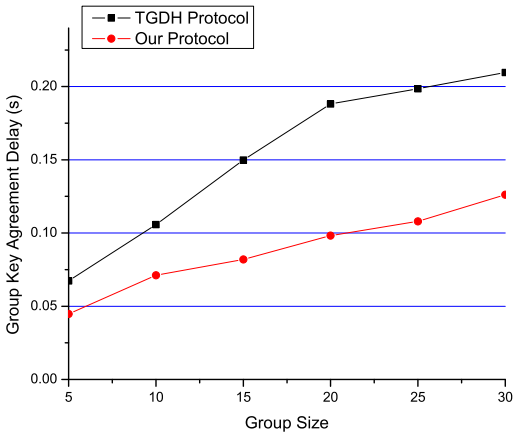
(c) Join for Network Size = 80

(d) Join for Network Size = 100

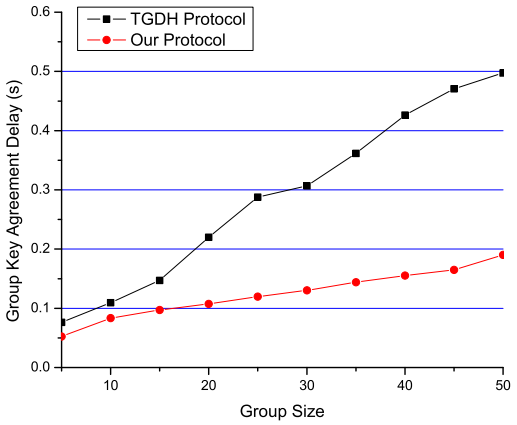
Figure 4.12: Join Delay for Different Network Sizes

Fig. 4.12 shows the group key agreement delay when a node joins the group for different networks sizes. As can be seen from the figures, the delay of our protocol is almost always less than that of the TGDH protocol. And as the group size increases, the

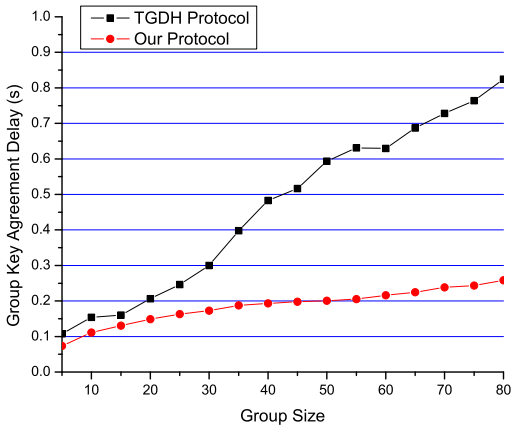
delay incurred by the TGDH protocol increases dramatically, while our protocol only causes a slowly increasing delay.



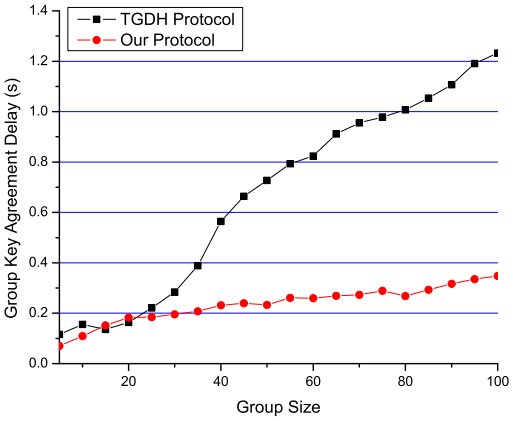
(a) Leave for Network Size = 30



(b) Leave for Network Size = 50



(c) Leave for Network Size = 80



(d) Leave for Network Size = 100

Figure 4.13: Leave Delay for Different Network Sizes

Similar to the join events, in Fig. 4.13 our protocol exhibits its efficiency over the TGDH protocol in the delay of leave events in group key agreement. For a small group size, our protocol and TGDH have similar delay for leave events. While as the group size becomes larger, the delay of TGDH increases very fast but our protocol only causes

moderate increase in delay for group members.

In our group key agreement protocol, we only consider the static network topology of ad hoc networks, and do not explicitly take the impact of mobility into consideration. Also the performance impact of mobility is not evaluated. However, we can expect a slightly dropdown on performance for small mobility and moderate mobility, since our group key agreement scheme is still efficient in message delivery.

4.5 Summary

In this Chapter, we presented a group key agreement scheme employing a novel key tree construction. The main contributions of our scheme can be summarized as follows:

- Our scheme constructs the key tree piggybacking on the multicast tree from the underlying multicast scheme. With this specially constructed key tree, the group key agreement can be accomplished with great efficiency.
- The efficiency in communications is greatly improved with our key tree. Constructed from the corresponding multicast tree, our key tree greatly facilitates transmission of key information within the group. As a result of communication efficiency, the throughput of the network is improved while the power consumption, delay, and the packet loss ratio are decreased greatly, which are important in ad hoc networks.
- It demonstrates striking advantage over other key trees regarding membership events. The process of our key tree construction embodies the network topology and this property reduces computation and communication efforts dramatically

for our key tree in join, leave, partition and merge operations.

- We implement our protocol on ns-2 and present the simulation result for the performance of our protocol. The result shows that our protocol has a much less delay compared with TGDH, and as the network size and the group size increase, the delay of our protocol increases very slowly while the delay of the TGDH protocol increases dramatically.

Furthermore, our key tree construction approach can be used to replace other key trees of the tree-based group key agreement protocols to improve their performance.

CHAPTER 5

Group Password-Authenticated Key Agreement Protocol for Infrastructured Multi-hop Wireless Networks

5.1 Introduction

Mobile ad hoc networks (MANET) still have not changed our way of using wireless networks. The ad hoc mode of 802.11 networks is seldom switched on by normal users, but is almost only used in laboratory testbeds. The reason that ad hoc networks have not been massively deployed is because current research and development of MANET are focusing on specialized applications (military applications, disaster recovery etc.). To make MANET a more commercialized commodity, *wireless mesh networks*(WMN), a new type of wireless networks emerges as an infrastructured multi-hop wireless network. Mesh networks provide a flexible and low-cost extension of wired infrastructure networks with ad hoc network technology, and hence they have advantages of both ad hoc networks and wireless LAN.

Except that mesh networks are connected to wired networks, mesh networks are almost the same as MANET and inherit many features of MANET. Mesh networks are multihop networks with dynamic network topology. Each node is still required to serve

as both a router and a normal mobile node, and users can dynamically join or leave the network. Wireless mesh networks have the following characteristics:

- WMN supports ad hoc networking, and is capable of self-organization. The network topology of WMN is dynamic and each node in WMN is still required to serve as both a router and a normal mobile node. Therefore, many research result can be applied on WMN as well.
- WMN is multi-hop wireless networks with a wireless infrastructure or backbone. Hence Internet access or trusted servers are possibly provided in WMN.

Although mesh networks are still new compared to MANETs, they have shown their strong potential in commercial applications. Community networks are a type of mesh networks that provide Internet access to a community of users that share the same Internet access link, and they can be used for neighborhood surveillance and emergency management. Mesh networks are also the ideal solution for intelligent transportation systems to alleviate transportation congestion and improve safety and security. Such a system has been implemented by Meshnetworks Inc. and the system has been deployed by public transportation companies.

Due to the proliferation of group-oriented applications, e.g. teleconferencing, collaborative workspaces, there is a rising demand for secure group key agreement in wireless mesh networks. As human-memorable passwords are extensively used for user authentication and key exchange in applications like internet banking, remote user access etc., a password-only group key agreement is preferable in wireless mesh networks.

The problem of authentication and key exchange between two parties sharing a password, referred to as the two-party *password-authenticated key exchange* (2PAKE)

problem, has been well studied and many solutions have been proposed in the literature. While the group PAKE protocol has not received enough research efforts.

In group oriented communications, either the group shares a single password, or each client in the group shares an independent password with a trusted server. The single-password setting is not preferable in real applications for several reasons. First, if a client in the group leaves or the password of a client is compromised, the shared password has to be updated, which could be a very expensive process. Moreover, compromise of any client leads to breakdown of the entire system. Secondly, individual client identification is impossible in this setting. As a result, no one is able to distinguish one client from another, and it is impossible for a subgroup of the group to securely establish a session key and have secure communications. It is easy to see that the independent-password setting avoids the above problems and reflects more accurately what is happening in the real world.

Group PAKE protocols in the independent-password setting need more careful treatment since they suffer from attacks which are not present in the single password setting, such as attacks initiated by legitimate clients against other clients' passwords. Not only should passwords be resistant to outsider attacks, but they should be secure against insider attacks.

In this Chapter, we propose an efficient group PAKE protocol, referred to as nPAKE$^+$ protocol, well-suited for wireless mesh networks under the independent-password setting. By employing a Diffie-Hellman key tree in group key establishment, the protocol achieves group key establishment with only 4 message flows, and every client needs only to perform $5 + \lceil \log n \rceil$ exponentiations. In our protocol, we do not explicitly taken into

account the impact of mobility, and its impact on performance is not evaluated.

5.2 Our nPAKE$^+$ Protocol for Multi-hop Wireless Networks

In this section, we present a group PAKE protocol, called nPAKE$^+$ protocol, under the independent-password setting for multi-hop wireless mesh networks.

Group key agreement problem using only passwords for wireless mesh networks is described as follows. As depicted in Fig. 5.1, clients in a typical wireless mesh network want to hold a secure group conference with the help of a trusted third server (not shown in the figure) located in Internet, and all they can use for group key agreement is a weak password each of them shared independently with the server.

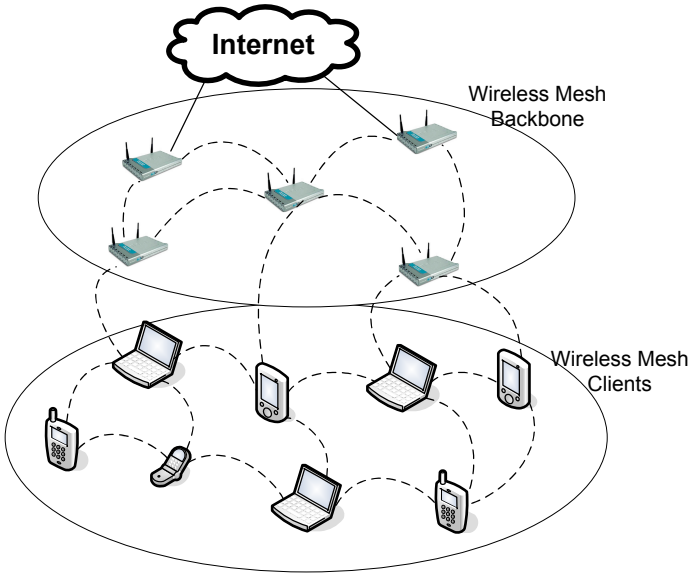


Figure 5.1: A Typical Topology of Mesh Networks

Our group PAKE protocol under the independent-password setting is well-suited

for wireless mesh networks for the following reasons. Infrastructure like trusted servers is available in mesh networks, and this meets the requirement for a trusted server of our protocol. Hop-by-hop message transmission style of our protocol matches multi-hop feature of wireless mesh networks perfectly. Our protocol also provides great efficiency in communications and computation, which is desired in bandwidth-limited wireless mesh networks.

5.2.1 System Setup and Requirements

In the system set-up of our nPAKE$^+$, n clients $C_1, C_2, ..., C_n$ share n independent passwords $p_1, p_2, ..., p_n$ with a trusted server S respectively. They agree on two large primes p and q with $p = 2q+1$, a subgroup $G_{p,q}$ of Z_p^*, a generator g of $G_{p,q}$ and a cryptographic secure keyed hash function $H_K(\cdot)$. Notations used in the description of the protocol are given in Table 5.1.

Table 5.1: Notations for Group PAKE Protocol

C_i	The i-th client, $i = 1, 2, ..., n$
S	The trusted server
p_i	The password shared between C_i and S
p, q	Two large primes with $p = 2q + 1$
$G_{p,q}, g$	The subgroup of order q in Z_p^* and its generator, respectively
$H_K(\cdot)$	A keyed cryptographically secure hash function using a key K
K_i, BK_i [1]	The secret key and blinded key for client $C_i, i = 1, 2, ..., n$
$\langle l, v \rangle$	The vth node at the lth level on the binary key tree, refer to Fig. 5.2
$K_{\langle l,v \rangle}, BK_{\langle l,v \rangle}$	The secret key and blinded key for node $\langle l, v \rangle$
$SK_{i,i+1}$	The session key shared between C_i and $C_{i+1}, i = 1, 2, ..., n-1$

[1] They are interchangeable with $K_{\langle l,v \rangle}, BK_{\langle l,v \rangle}$ if C_i is located at $\langle l, v \rangle$ on the key tree.

A group PAKE protocol should meet the following security requirements:

- Mutual Authentication: Each client should have mutual authentication with the server to thwart attacks like masquerade attacks.

- Password Secrecy: Passwords should be protected from off-line dictionary attacks, and online password guessing attacks should be detected.
- Group Key Secrecy: The established group key is only shared among group members, not known by any entity outside the group.
- Perfect Forward Secrecy: Compromise of the long-term secret passwords does not lead to disclosure of previous group keys.

5.2.2 The Diffie-Hellman Key Tree

Key graphs are extensively used in (non-password based) group key agreement protocols to achieve great efficiency in computation and communications. Wong et al. [70] and Wallner et al. [73] are the first to introduce the concept of a key graph, called the *Logical Key Hierarchy*(LKH), to improve efficiency in group key management. The *One-way Function Tree*(OFT) proposed by McGrew and Sherman [69] improves the hierarchical tree approach further. In the OFT, the key of the parent is derived from the keys of its children, and hence it reduces the size of the rekeying messages to half of the LKH. Based on the key tree, some group key agreement proposals [74, 76, 102, 117, 120, 121] used the Diffie-Hellman exchange technique in group key establishment.

The Diffie-Hellman key tree used in our protocol has the same structure as that discussed in Section 4.2.1, with minor differences. It is also a binary tree on which each leaf represents a group member. Every interior node of the key tree has exactly two leaves, and is not associated with any group member. An example of the key tree used in our protocol is illustrated in Fig. 5.2. The same notations used in Section 4.2.1 are used here. The nodes are denoted $\langle l, v \rangle$, where $0 \leqslant v \leqslant 2^l - 1$ since each level l hosts at

most 2^l nodes (the root is at the 0-th level). For any interior node $\langle l, v\rangle$, its left child and right child are denoted $\langle l+1, 2v\rangle$ and $\langle l+1, 2v+1\rangle$ respectively. Each node $\langle l, v\rangle$ on the key tree is associated with a secret key $K_{\langle l,v\rangle}$ and a corresponding blinded key $BK_{\langle l,v\rangle}$ computed as $g^{K_{\langle l,v\rangle}} \mod p$.

The only difference in this key tree structure is the secret key at the leaf node. The secret key $K_{\langle l,v\rangle}$ at a leaf node $\langle l, v\rangle$, which is associated with a client C_i, is constructed between the client C_i and the server S in our protocol. While the secret key of an interior node is derived from the keys of the interior node's two children by Diffie-Hellman computation. The corresponding blinded key is then computed following the formula $BK_{\langle l,v\rangle} = g^{K_{\langle l,v\rangle}} \mod p$. Specifically, the secret key and the blinded key of an interior node $\langle l, v\rangle$ are computed recursively as follows:

$$\begin{aligned} K_{\langle l,v\rangle} &= g^{K_{\langle l+1,2v\rangle} K_{\langle l+1,2v+1\rangle}} \mod p \\ &= (BK_{\langle l+1,2v\rangle})^{K_{\langle l+1,2v+1\rangle}} \mod p \\ &= (BK_{\langle l+1,2v+1\rangle})^{K_{\langle l+1,2v\rangle}} \mod p, \\ BK_{\langle l,v\rangle} &= g^{K_{\langle l,v\rangle}} \mod p. \end{aligned}$$

Note that if a client C_i is located at a leaf node $\langle l, v\rangle$ on the key tree, then its secret key and blinded key $K_{\langle l,v\rangle}, BK_{\langle l,v\rangle}$ are also denoted as K_i and BK_i respectively. These two types of denotations (see Fig. 5.2) are interchangeable for a client C_i at a leaf node $\langle l, v\rangle$.

Therefore, computing a secret key at $\langle l, v\rangle$ requires the knowledge of the key of one child and the blinded key of the other child. The secret key $K_{\langle 0,0\rangle}$ at the root node is

the group key which should be known only to the group members.

In order to compute the group key, a member C_i needs to know a set of blinded keys, which form a set called the co-path. With the blinded keys in the co-path, the member C_i can compute a set of keys from itself to the root of the key tree, and these keys form another set called key-path. For a client C_i located at a leaf node $\langle l, v\rangle$, we denote its key-path as KP_i or $KP_{\langle l,v\rangle}$, its co-path as CP_i or $CP_{\langle l,v\rangle}$. On the key tree, the key path KP_i is a path from C_i itself until the root node ($\langle 0, 0\rangle$) of the key tree. While the co-path CP_i is formed by all the nodes that are directly connected with the key-path KP_i on the key tree. And the key-path KP_i splits the co-path CP_i into two halves: R_i on the right side and L_i on the left side.

For example, in Fig. 5.2 the client C_2's key-path is $KP_2 = KP_{\langle 3,1\rangle} = \{K_{\langle 3,1\rangle}, K_{\langle 2,0\rangle}, K_{\langle 1,0\rangle}, K_{\langle 0,0\rangle}\}$, and its co-path is $CP_2 = CP_{\langle 3,1\rangle} = \{BK_{\langle 3,0\rangle}, BK_{\langle 2,1\rangle}, BK_{\langle 1,1\rangle}\}$. The key-path KP_2 is a path from C_2 (or $\langle 3, 1\rangle$) until the root of the key tree. And each node from the co-path CP_2 is directly connected with the key-path KP_2 on the key tree. The co-path CP_2 is split into two halves by the key-path KP_2: $R_2 = \{BK_{\langle 2,1\rangle}, BK_{\langle 1,1\rangle}\}$, and $L_2 = \{BK_{\langle 3,0\rangle}\}$.

The following two properties of the key tree are important for group key agreement in our protocol:

- For any binary Diffie-Hellman key tree with n leaves labelled from C_1 to C_n, client C_i can compute L_{i+1} using L_i, K_i, and $\{BK_j : 1 \leq j \leq n\}$. Similarly, C_i can compute R_{i-1} using R_i, K_i, and $\{BK_j : 1 \leq j \leq n\}$.
- For any binary Diffie-Hellman key tree with n leaves labelled from C_1 to C_n, client C_i can compute the group key using L_i, R_i, and K_i.

The second property is clear from the definition of the co-path since the co-path $CP_i = L_i \cup R_i$ is defined as the set of blinded keys from which the group key can be computed by client C_i. For the first property of the key tree, we decribe how C_i can compute R_{i-1} using R_i, K_i, and $\{BK_j : 1 \leq j \leq n\}$. Suppose client C_i and C_{i-1} locate at node v_i and v_{i-1} on the key tree, respectively. Let u be the nearest common ancestor of v_i and v_{i-1}, and u_l and u_r are u's left and right child respectively. Then it is clear that $R_{i-1} \subseteq R_i \cup BK_{v_r}$ where BK_{v_r} denotes the blinded key at node v_r. For the subtree that has v_r as the root, v_i is the leftmost node of the subtree. Hence client C_i can compute BK_{v_r} using R_i and K_i, and R_{i-1} can then be deduced from R_i and BK_{v_r}.

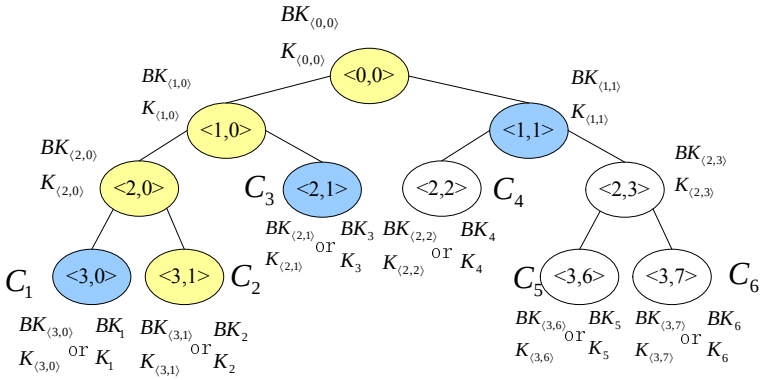


Figure 5.2: An Example of the Key Tree

With all the blinded keys of its co-path, a client C_i can compute all the keys along the key-path, including the group secret $K_{\langle 0,0\rangle}$. For the example in Fig. 5.2, with its own key $K_{\langle 3,1\rangle}$, C_2 can compute $K_{\langle 2,0\rangle}$, $K_{\langle 1,0\rangle}$ and $K_{\langle 0,0\rangle}$ using $BK_{\langle 3,0\rangle}$, $BK_{\langle 2,1\rangle}$, $BK_{\langle 1,1\rangle}$, respectively.

5.2.3 Description of the Protocol

After introducing the Diffie-Hellman key tree, we describe our nPAKE$^+$ protocol in this section. Our protocol achieves group key establishment and authentication with 3 message flows. The first flow starts from the client C_1, traverses through $C_2, C_3, ..., C_n$, and finally reaches the server S. After that, the second flow initiated by the server in the reverse direction from S until C_1. After the second flow terminates at C_1, C_1 starts the third flow towards C_n, and the third flow stops at S.

- **Flow 1:** Client C_1 chooses $r_1 \in_R Z_q$ and initiates the first flow by sending a request comprising of the identities $\{C_i\}_{i=1}^n$ of all clients to join the group and an exponential $(g^{r_1})_{p_1}$ encrypted with its password. The request traverses all the clients (from C_1 to C_n) until it reaches the server. Upon receiving the request, each client C_i selects $r_i \in_R Z_q$ and adds his own encrypted exponential $(g^{r_i})_{p_i}$ into the request. So when the request finally reaches the server S, the request consists of n identities and n encrypted exponentials contributed by the n clients.

$$\begin{aligned} C_i \longrightarrow C_{i+1} \quad &: \quad \{C_j\}_{j=1}^n | \{(g^{r_j})_{p_j}\}_{j=1}^i, \quad i = 1, 2, ..., n-1, \\ C_n \longrightarrow S \quad &: \quad \{C_j\}_{j=1}^n | \{(g^{r_j})_{p_j}\}_{j=1}^n. \end{aligned} \tag{5.1}$$

- **Flow 2:** The second flow of messages runs in the reverse direction, from the server S to C_1. After receiving the request, the server parses $\boldsymbol{\{C_i\}_{i=1}^n | \{(g^{r_i})_{p_i}\}_{i=1}^n}$, and uses corresponding passwords to decrypt $(g^{r_i})_{p_i}$ to obtain $g^{r_i} (i = 1, 2, ..., n)$. Then for each client $C_i (i = 1, 2, ..., n)$, S chooses $s_i \in_R Z_q$ and computes a session key $K_i = (g^{r_i})^{s_i}$. Then the server computes $\pi = BK_1 | C_1 | \cdots | BK_n | C_n$ and $\tau_i =$

$H_{K_i}(\pi)$, and sends $\boldsymbol{\pi|\{(g^{s_j})_{p_j}|\tau_j\}_{j=1}^{n}}$ to C_n.

The reply originated from the server S passes through C_n to C_1. Receiving the reply, C_i parses it as $\boldsymbol{\pi|\{(g^{s_j})_{p_j}|\tau_j\}_{j=1}^{i}|R_i|\xi_{i,i+1}}$. By default, $R_n|\xi_{n,n+1} = nil$. C_i decrypts $(g^{s_i})_{p_i}$ to obtain g^{s_i} using his password. Then he computes the session key $K_i = (g^{s_i})^{r_i}$ and the blinded $BK_i = g^{K_i}$, and verifies whether the computed BK_i equals to BK_i in π. Then C_i verifies the validity of π by checking $H_{K_i}(\pi) \stackrel{?}{=} \tau_i$. In the case where $i \neq n$, C_i also computes $SK_{i,i+1} = (BK_{i+1})^{K_i}$ and verifies R_i by checking whether $H_{SK_{i,i+1}}(R_i|0)$ equals $\xi_{i,i+1}$.

If the reply passes all verifications, $C_i(i = 2, 3, ..., n)$ prepares an outgoing message for the next client C_{i-1}. C_i computes R_{i-1} with R_i, K_i and π, and computes $SK_{i-1,i} = (BK_{i-1})^{K_i}$. Then he computes $\xi_{i-1,i} = H_{SK_{i-1,i}}(R_{i-1}|0)$ and sends $\boldsymbol{\pi|\{(g^{s_j})_{p_j}|\tau_j\}_{j=1}^{i-1}|R_{i-1}|\xi_{i-1,i}}$ to C_{i-1}.

$$\begin{aligned} S &\longrightarrow C_n \quad : \quad \pi|\{(g^{s_j})_{p_j}|\tau_j\}_{j=1}^{n}, \\ C_i &\longrightarrow C_{i-1} \quad : \quad \pi|\{(g^{s_j})_{p_j}|\tau_j\}_{j=1}^{i-1}|R_{i-1}|\xi_{i-1,i}, \;\; i = n, ..., 2. \end{aligned} \tag{5.2}$$

where $\pi = BK_1|C_1|\cdots|BK_n|C_n$.

- **Flow 3:** When the reply in Flow 2 finally reaches C_1, C_1 does the verifications as specified in Flow 2. If verifications are successful, then C_1 computes the group key with R_1 and K_1 as well as π. Then C_1 computes L_2, $\sigma_{1,2} = H_{SK_{1,2}}(L_2|1)$, $\eta_1 = H_{K_1}(C_1|C_2|\cdots|C_n)$, and starts the last flow by sending out $L_2|\sigma_{1,2}|\eta_1$ to C_2.

  Then each client $C_i(i = 2, 3, ..., n)$ receives the message $L_i|\sigma_{i-1,i}|\{\eta_j\}_{j=1}^{i-1}$, and he verifies L_i by checking $\sigma_{i-1,i} \stackrel{?}{=} H_{SK_{i-1,i}}(L_i|1)$. If the verification is suc-

cessful, he computes the group key with K_i, L_i, R_i and π. If $i \neq n$, C_i computes $\sigma_{i,i+1} = H_{SK_{i,i+1}}(L_{i+1}|1)$, computes L_{i+1} from L_i, K_i and π, computes $\eta_i = H_{K_i}(C_1|C_2|\cdots|C_n)$, and sends the outgoing message $L_{i+1}|\sigma_{i,i+1}|\{\eta_j\}_{j=1}^{i}$ to C_{i+1}. Otherwise, C_n computes η_n and sends $\{\eta_j\}_{j=1}^{n}$ to the server S.

$$\begin{aligned} C_i &\longrightarrow C_{i+1} \quad : \quad L_{i+1}|\sigma_{i,i+1}|\{\eta_j\}_{j=1}^{i}, \quad i = 1, ..., n-1. \\ C_n &\longrightarrow S \quad : \quad \{\eta_j\}_{j=1}^{n} \end{aligned} \tag{5.3}$$

After the third flow finally reaches the server, the server verifies each η_i from client C_i to authenticate each client. If any verification is failed, then the server can identify which client(s) is(are) invalid and not authenticated. This measure is intended to thwart on-line password guessing attacks.

After the last flow reaches the server, each client has already computed his own L_i and R_i, so each client obtain his own co-path $CP_i = L_i \cup R_i$. Therefore, every client can independently calculate the same group key $K_{\langle 0,0 \rangle}$ and use it for secure group communications.

5.3 Security and Performance Analysis

In this section, we provide an *informal* analysis of security for our protocol, and evaluate the performance of our protocol afterwards. We are going to show the property of group key secrecy, clients' password secrecy, and perfect forward secrecy of our protocol. In the analysis, we assume that the server is not corrupted as corrupting it can trivially compromise the system.

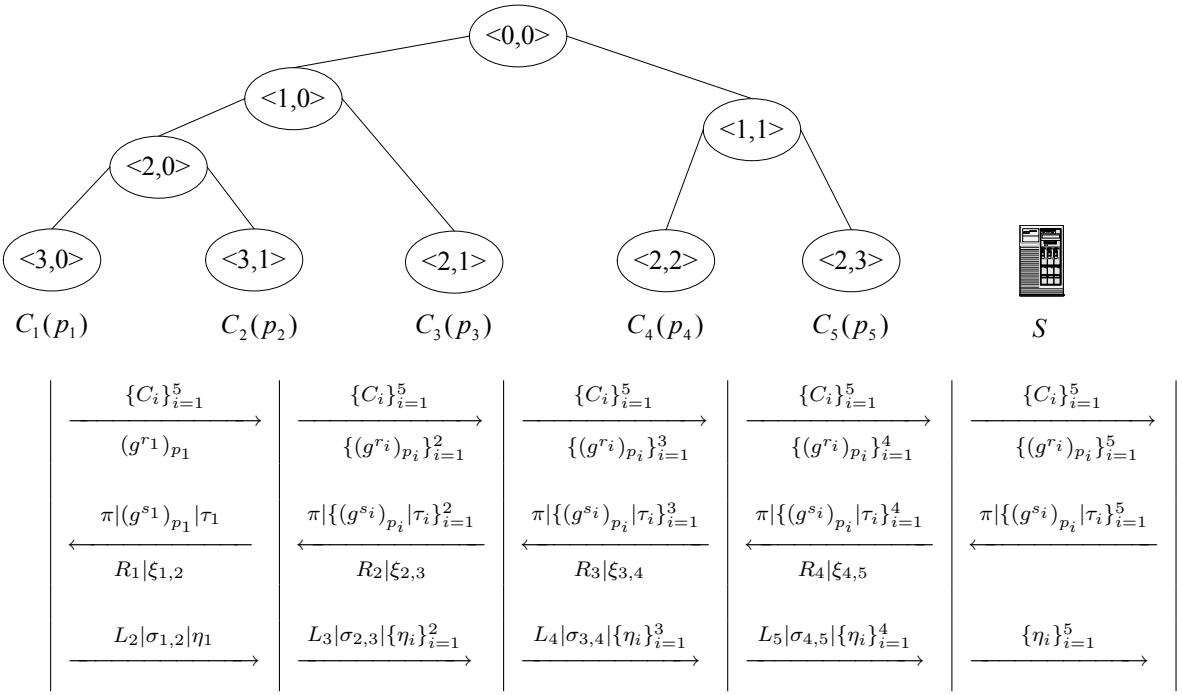


Figure 5.3: An Example of the Protocol with 5 Nodes

Mutual Authentication. Mutual authentication between each client and the server is required to thwart possible attacks like masquerade attacks. Under the independent-password setting, only the server knows the password for each client and it is the server's responsibility to authenticate legitimate clients. At the same time, each client needs to authenticate the server to ensure the server is not cheating on him.

In the second flow of our protocol, the server computes the session key K_i with each client C_i, and sends $\pi = BK_1|C_1|\cdots|BK_n|C_n$ and $\tau_i = H_{K_i}(\pi)$ back to clients. So each client is able to authenticate the server by checking $H_{K_i}(\pi) \stackrel{?}{=} \tau_i$. This verification also ensures BK_i is not modified by any client.

While in the third flow of our protocol, each client also computes the session key K_i and sends the server $\eta_i = H_{K_i}(C_1|C_2|\cdots|C_n)$. The server then can authenticate each

client by verifying $H_{K_i}(C_1|C_2|\cdots|C_n) \stackrel{?}{=} \eta_i$.

Group Key Secrecy. In our protocol, the server is authenticated to each client and each client is also authenticated to the server independently. Therefore, only legitimate parties can finish the protocol and derive the correct group key.

After the server receives the first flow of messages, it establishes n session keys $K_i(1 \leq i \leq n)$ with n clients respectively. Then the server sends back the reply (see Flow 2) $\boldsymbol{\pi|\{(g^{s_j})_{p_j}|\tau_j\}_{j=1}^{n}}$ to C_n. Upon receiving the reply, a client C_i decrypts $(g^{s_i}|\tau_i)_{p_i}$ and computes K_i and BK_i. The client authenticates the server by checking whether the computed BK_i equals the one in π. Only if the server is authentic can it be authenticated by the clients. If the authentication is successful, the client checks whether other blinded keys in π are valid by verifying $\tau_i \stackrel{?}{=} H_{K_i}(\pi)$. If the message passes the verification, the client $C_i(i = n-1, n-2, ..., 1)$ can authenticate C_{i+1} and R_i by checking whether $H_{SK_{i,i+1}}(R_i|0) \stackrel{?}{=} \xi_{i,i+1}$. In the third flow of the protocol, client $C_i(i = 2, 3, ..., n)$ authenticates client C_{i-1} and L_i by checking $\sigma_{i-1,i} \stackrel{?}{=} H_{SK_{i-1,i}}(L_i|1)$. Whereafter each client C_i can compute the group key $K_{\langle 0,0\rangle}$ using R_i, L_i and his own session key K_i. When the third flow reaches the server, the server authenticates each client by checking $\eta_i \stackrel{?}{=} H_{K_i}(C_1|C_2|\cdots|C_n)$. Therefore, each client are mutually authenticated with the server, and a secret key K_i is established between client C_i and the server. The group key $K_{\langle 0,0\rangle}$ is constructed in a contributory way from the authenticated shares of all valid clients, so it can only be correctly derived by valid group members.

Password Secrecy. For a passive outside attacker who eavesdrops the communications between the clients and the server, it is computationally infeasible for him to discover clients' passwords or the secret group key. If the adversary attempts to launch

a dictionary attack to client C_i's password , he first guesses a password p_i', uses p_i' to decrypt $(g^{r_i})_{p_i}$ and $(g^{s_i})_{p_i}$ to obtain $g^{r_i'}$ and $g^{s_i'}$; however, he needs to compute $g^{r_i's_i'}$ to verify whether his guess is correct, which is infeasible assuming the hardness of the Diffie-Hellman problem. If the adversary's objective is to find the secret group key, he needs to know at least one client's session key K_i to recover the group key. Since he even cannot compromise the passwords to obtain g^{r_i} and g^{s_i}, he is unable to derive the correct $K_i = g^{r_i s_i}$. As a result, the adversary has no way to discover the secret group key.

An active outside attacker impersonating a client or the server would not be authenticated successfully by the server or other clients, and it would be detected. Not having the correct password, the attacker cannot compute the correct K_i and authenticate itself to the server. Furthermore, to be authenticated successfully by another client, client C_i must be able to compute session keys $SK_{i-1,i}(i = 2, 3, ..., n)$ and $SK_{i,i+1}(i = 1, 2, ..., n-1)$. As discussed earlier, the attacker is unable to obtain the correct K_i, and hence he cannot derive the correct session keys $SK_{i-1,i}$ and $SK_{i,i+1}$.

An outside adversary may also impersonate as a group of clients to trick the server to run the protocol with them. In this case, the adversary can impersonate n clients and guest one password for each client in a single protocol run. This on-line password-guessing attack is a serious threat for the protocol, and it is important for the server to detect the attack to thwart such attacks. In the third flow of our protocol, a client C_i needs to compute η_i to authenticate himself to the server. If he fails to authenticate himself to the server, then the server will detect the attack and continuous failure will alert the server.

It is possible that one client is interested in other clients' passwords. We further claim that in our protocol no client is able to discover other clients' passwords. A malicious client can only decrypt the exponential encrypted with his own password, but not those exponentials encrypted with other clients' passwords. For any valid client in the protocol, As a result, the malicious client does not have any advantage over an outside attacker. As analyzed before, it is computationally infeasible for a malicious client to disclose other clients' passwords by launching any attacks including dictionary attacks.

Perfect Forward Secrecy. Since the group key is contributorily constructed by all the clients and the server using Diffie-Hellman exchange, past group keys are secure against compromise of the long-term secret passwords, and hence perfect forward secrecy is achieved in our protocol.

Performance. Under the independent password setting, our protocol is both flexible and efficient in communications and computation. First, under the independent password setting, our protocol accommodates formations of secure subgroups. Any subgroup of the whole group can run the protocol to establish a group key for the subgroup. Secondly, the protocol employs key tree in group key construction to provide communication and computation efficiency. This greatly reduces communication and computation costs. The protocol needs only three message flows to establish the group key, and each client needs only $5 + \lceil \log n \rceil$ exponentiations while the server needs $3n$ exponentiations. A comparison on computation efficiency between the protocol under the single password setting by Bresson *et al.* [101], the protocol EKE-U by Byun and Lee [95] and our protocol is given in Table 5.2. Bresson's protocol requires $(n + 5)/2$

exponentiations for each client on average, EKE-U protocol requires $(n+3)/2$ exponentiations for each client, while our protocol requires only $5+\lceil \log n \rceil$ for each client at most. Though our protocol requires the server to complete $3n$ exponentiations additionally, the total number of exponentiations required in our protocol $(n(8+\lceil \log n \rceil))$ is still much lower than that required in Bresson's protocol and EKE-U protocol.

Furthermore, each client shares a password with the server instead of sharing pairwise secrets with all other clients, hence the protocol scales to large group size.

Table 5.2: Computation and Communication Cost Comparison between Group Password-based Protocols

	Client (Avg.)	Server	Total	No. of Msgs
Bresson's Protocol	$(n+5)/2$	-	$n(n+5)/2$	$3n(n^2/2)$[1]
EKE-U Protocol	$(n+3)/2$	$(n+1)(n+2)/2$	n^2+3n	$4n(3n^2/2)$
Our Protocol	$5+\lceil \log n \rceil$	$3n$	$n(8+\lceil \log n \rceil)$	$3n(n^2)$

[1]Numbers in brackets give total message size by taking the exponential size as a unit.

5.4 Summary

In this Chapter, we proposed a server-assisted group password-authenticated key exchange protocol where each client shares an independent password with a trusted server. Under this independent-password setting, our protocol provides better flexibility than those protocols under the single-password setting. Moreover, the protocol employs a Diffie-Hellman key tree for group key agreement, and hence achieves great efficiency in both computation and communications. Finally, we provided a detailed security and performance analysis for the proposed protocol. Our future work is to give a formal security proof for the protocol.

CHAPTER 6

Conclusions and Future Research

Conclusions

Popularity and extensive applications of wireless communications bring many benefits like convenience, flexibility for users. Due to the unique characteristics of wireless networks, such networks also bring new challenges in security issues as well as known security risks in wired networks. In wireless networks, open transmission medium, limited power, restricted computation capability and network bandwidth make it very difficult to design satisfactory security protocols for such networks. Resource constraints make wireless networks more vulnerable to denial of service attacks than wired networks. Mobility in wireless networks also introduces new requirements in security protocol design. As mobile users are roaming within wireless networks, users' private information like location, movement pattern should be protected from potential adversaries.

While group key management in wireless ad hoc networks is further complicated by complexity of multi-party protocols and absence of infrastructure in such networks. Multi-party security protocols is much more complex than two-party ones as more weak links exist in group key protocols and more attacks are possible to compromise the system. What makes things worse is lack of infrastructure in ad hoc networks. Since there is no authority or trusted server existing in ad hoc networks, mobile users are

required to authenticate each other and agree on a group key on their own. Moreover, group key management schemes need to consider not only group key establishment but also membership dynamics.

In this thesis, we investigated authentication and key establishment in wireless networks. We first studied two-party authentication and key exchange problems in wireless networks, and analyzed previous solutions for wireless LAN. We identified security weaknesses in previous security schemes for wireless networks, and proposed two new protocols to replace them. Our first authentication and key exchange protocol is based on public key cryptosystem. While the other protocol employs a weak password shared between a client and a server to achieve authentication and key exchange. The PKC-based protocol requires both the client and the server have a certificate. While our password-based protocol does not require a certificate on the client side but a shared password between the client and the server. Both our protocols achieve mutual authentication and secure key exchange for access control in wireless networks, and they also offer client identity anonymity and resistance to DoS attacks. Since the password-based protocol removes the need for client certificates, it gets rid of burden in certificate management and hence provides great convenience for clients.

We then studied group key agreement for ad hoc networks. Most of previous proposals on group key agreement employ a binary key tree to improve computation and communication efficiency. But these schemes did not take network topology into account in protocol design. Therefore, we presented a group key agreement scheme constructing key tree from the underlying network topology. Our scheme constructs the key tree piggybacking on the multicast tree from the multicast scheme. With this specially con-

structed key tree, the group key agreement can be accomplished with great efficiency. This property reduces computation and communication efforts dramatically for our key tree in join, leave, partition and merge operations. We implemented our protocol on ns-2 and analyzed the performance of our protocol. The result shows that our protocol has a much less delay compared to TGDH, and as the network size and the group size increase, the delay of our protocol increases slowly while the delay of the TGDH protocol increases dramatically.

We also presented another group key agreement based on shared passwords : server-assisted group password-authenticated key exchange protocol. This protocol is efficient in both computation and communication, and it can be used in variants of ad hoc networks where trusted servers are available. In this protocol, each client shares an independent password with a trusted server, which is referred to as the independent-password setting. Under this independent-password setting, our protocol provides better flexibility than those protocols under the single-password setting. Moreover, the protocol employs a Diffie-Hellman key tree for group key agreement, and hence achieves great efficiency in both computation and communications. We also provided a detailed security and performance analysis for the proposed protocol.

Future Research Directions

Though security issues in wireless networks have attracted considerable attention and research efforts, there are still many challenging security problems in wireless networks needing to be solved. First of all, it is crucial to provide privacy protection in wireless

networks to thwart traffic analysis, movement tracing etc. Privacy issues for wireless networks comprise of identity anonymity, location privacy, routing privacy, network topology privacy, motion pattern privacy, to name but a few. However, relatively little work on anonymity [124] has been carried out in this direction for wireless networks, while other areas such as like unlinkability and unobservability [125] remain relatively untouched. On top of this we need to consider the possibility of providing DoS resistance at the same time.

- *Identity Anonymity.* Identity anonymity is a basic requirement in the prevention of privacy information such as location and motion pattern from being disclosed to adversaries. Anonymity can also ensure that the behavior of a mobile node is completely hidden from attackers. In a wireless environment, anonymity compromise also means compromise of location privacy and disclosure of motion patterns. How to design a strong and efficient anonymous scheme for ad hoc networks and sensor networks still remains a challenge.
- *Network Topology Privacy.* The mobility of wireless networks results in a dynamic network topology, and the network topology itself becomes a part of the privacy information that potentially needs protection from adversaries.
- *Location and Motion Pattern Privacy.* Advances in positioning technologies enable location-based services like GPS, and such location-based services provide convenience, safety and other benefits. While location information is used in designing secure schemes for routing or key agreement in a wireless environment. Consequently, protection of location privacy and motion pattern from being revealed has become a serious problem that needs solving.

- *Unlinkability and Unobservability.* Unlinkability and unobservability are two strong requirements for privacy protection. Though similar schemes have been proposed for wired networks, it cannot be employed for wireless networks directly due to the unique characteristics of wireless networks. We plan to achieve unlinkability and unobservability for ad hoc and sensor networks by adapting mechanisms used for wired networks.

Regarding group key management schemes, the concept of privacy protection is complicated and enriched by complexity of multi-party settings. It is a challenging task to design a sound group key management scheme to satisfy the privacy requirement for wireless networks. In group key management schemes, attacks against group members take much more complex forms than two-party protocols. A valid group member may want to probe information of another group member, while several group members could collude to compromise another member or the whole system. Moreover, privacy protection under multi-party settings means no one know who is in the group and who is not in the group. Also the factor of mobility should be considered when designing group key management schemes.

Bibliography

[1] W. Aliello et al., "Just Fast Keying (JFK)," *IETF Draft(work in progress)*, draft-ietf-ipsec-jfk-04.txt, July 2002.

[2] W. Aliello et al., "Efficient,DoS-Resistant,Secure Key Exchange for Internet Protocols," in *Proceedings of ACM Conference on Computer and Communication Security*, November 2002.

[3] Advanced Security for Personal Communications Technologies, available: http://www.esat.kuleuven.ac.be/cosic/aspect/

[4] A. Aziz and W. Diffie, "Privacy and Authentication for Wireless Local Area Networks," *IEEE Personal Communications,* First Quarter:25-31, 1994.

[5] K. Aoki and H. Lipmaa, "Fast Implementations of AES Candidates," *Third AES Candidate Conference*, New York City, USA, 13–14 April 2000.

[6] M. Aydos, B. Sunar and Ç.K. Koç, "An Elliptic Curve Cryptography based Authentication and Key Agreement Protocol for Wireless Communication," in *Proceedings of the 2nd International Workshop on Dicrete Algorithms and Methods for Mobility (DIALM'98)*, October 1998.

[7] W. A. Arbaugh, N. Shankar, and J. Wang, "Your 802.11 Networks Has No Clothes," in *Proceedings of the First IEEE International Conference on Wireless LANs and Home Networks*, December 2001.

[8] E. Bresson, O. Chevassut, and D. Pointcheval, "Proofs of Security for Password-Based Key Exchange (IEEE P1363 AuthA Protocol and Extensions)," *Cryptology ePrint Archive: Report 2002/192*, December 2002.

[9] M.J. Beller, L.-F. Chang, and Y. Yacobi, "Privacy and Authentication on a Portable Communications System," *IEEE Journal on Selected Areas in Communications*, 11:821-829, 1993.

[10] S.M. Bellovin,"Problem Areas for the IP Security Protocols," in *Proceedings of the 6th USENIX Security Symposium*, San Jose, California, July 1996.

[11] N. Borisov, I. Goldberg, and D. Wagner, "Intercepting Mobile Communications: The Insecurity of 802.11," in *Proceedings of the th 7th Annual International Conference on Mobile Computing and Networking*, July 16-21, 2001.

[12] C. Boyd and D.-G. Park, "Public Key Protocols for Wireless Communications," in *Proceedings of the 1998 Internatinal Conference on Information Security and Cryptology (ICISC'98)*, 1998.

[13] U. Carlsen, "Optimal Privacy and Authentication on a Portable Communications System," *ACM Operating Systems Review*, 28(3):16-23, 1994.

[14] T. Clancy and W. Arbaugh, "EAP Password Authenticated Exchange," *IETF Draft*, draft-clancy-eap-pax-04.txt, June 2005.

[15] D. Denning and G. Sacco, "Timestamps in key distribution protocols," *Communications of the ACM*, 24(8):533-536, 1981.

[16] J. Carlson, B. Aboba, H. Haverinen, "PPP EAP SRP-SHA1 Authentication Protocol," Internet-draft (work in progress), draft-ietf-pppext-eap-srp-03.txt, July 2001.

[17] P. Funk, "EAP Tunneled TLS Authentication Protocol Version 1," *IETF Draft*, draft-funk-eap-ttls-v1-00.txt, February 2005.

[18] B. Aboba and D. Simon, "PPP EAP TLS Authentication Protocol," *IETF RFC 2716*, October 1996.

[19] S. Josefsson et al., "Protected EAP Protocol (PEAP)," *IETF Draft*, draft-josefsson-pppext-eap-tls-eap-10 (work in progress), October 2004.

[20] Cisco, "Cisco LEAP Protocol Description," available: http://www.missl.cs.umd.edu/wireless/ ethereal/leap.txt, September 2001.

[21] J. Wright, "Asleap Homepage", [Online], available: http://asleap.sourceforge.net/

[22] D. B. Faria and D. R. Cheriton, "Dos and Authentication in Wireless Public Access Networks," in *Proceedings of ACM Workshop on Wireless Security*, September 2002.

[23] A. Friday et al., "Network Layer Access Control for Context-Aware IPv6 Applications," *Wireless Networks*, 9(4):299-309, 2003.

[24] Y. H. Hwang, D. H. Yum, and P. J. Lee, "EPA: An Efficient Password-Based Protocol for Authenticated Key Exchange," *ACISP 2003*, LNCS 2727, Springer-Verlag Berlin Heidelberg 2003, pages 452-463, 2003

[25] C. Kaufman, "Internet Key Exchange (IKEv2) Protocol," *IETF Draft (work in progress)*, draft-ieft-ipsec-ikev2-14.txt, June 2004.

[26] D. P. Jablon, "Strong Password-Only Authenticated Key Exchange," *ACM Computer Communications Review*, 26(5):5-26, October 1996.

[27] S. Kent and R. Atkinson, "IP Authentication Header," *IETF Standards Track RFC 2402*, November 1998.

[28] P. Karn and W. Simpson, "Photuris: Session-Key Management Protocol," *IETF RFC 2522*, 1999.

[29] T. Kwon, "Authentication and Key Agreement via Memorable Password," in *Proceedings of the ISOC NDSS Symposium*, 2001.

[30] A. Mishra and W. A. Arbaugh, "An Initial Security Analysis of the IEEE 802.1X Standard," Technical Report CS-TR-4328, UMIACS-TR-2002-10, University of Maryland, February 2002.

[31] W. Mao and C.H.Lim, "Cryptanalysis in Prime Order Subgroups of $\mathbb{Z}_n^*$," Advances in Cryptology-ASIACRYPT'98, LNCS 1514, Spinger-Verlag, 1998, pp.214-226.

[32] Y. Mu and V. Varadharajan, "On the Design of Security Protocols for Mobile Communications," *Information Security and Privacy*, Lecture Notes in Computer Science, 1172:134-145, 1996.

[33] LAN MAN Standards Committee of the IEEE Computer Socciety, "Wireless LAN medium access control (MAC) and physical layer (PHY) specification," *IEEE Standard 802.11,1997 Edition*, 1997.

[34] LAN MAN Standards Committee of the IEEE Computer Socciety, "Standard for Port Based Network Access Control," *Technical Report Draft P802.1X/D11*, IEEE Computer Society, March 2001.

[35] LAN MAN Standards Committe of the IEEE Computer Society, "Amendment 6: Medium Access Control (MAC) Security Enhancements," *IEEE Standards P802.11i*, June 2004.

[36] P. MacKenzie, "More Efficient Password-Authenticated Key Exchange," *Progress in Cryptology – CT-RSA 2001*, pages 361-377, 2001.

[37] S. Patel, "Number Theoretic Attacks on Secure Password Schemes," in *Proceedings of IEEE Symposium on Research in Security and Privacy*, pages 236-247, 1997.

[38] B. Preneel et al., "Performance of Optimized Implementations of the NESSIE Primitives," *NESSIE Report*, Delivrable D12, February 2003.

[39] S. Schmid et al., "An Access Control Architecture for Microcellular Wireless IPv6 Networks," in *Proceedings of 26th IEEE Conference on Local Computer Networks (LCN'2001)*, pages 454-463, 2001.

[40] M. Scott, "Multiprecision Integer and Rational Arithmetic C/C++ Library ," Shamus Software Ltd, available: http://indigo.ie/∼mscott/

[41] P. C. van Oorschot and M. Wiener, "On Diffie-Hellman Key Agreement with Short Exponents," *Eurocrypt'96* LNCS 1070, pages 332-343, 1996.

[42] Z. Wan and S. Wang, "Cryptanalysis of Two Password-Authenticated Key Exchange Protocols," in *Proceedings of Australasian Conference on Information Security and Privacy*, July 2004.

[43] W. Dai, "Crypto++ 5.1 Benchmarks," available: http://www. eskimo.com/weidai/benchmarks.html

[44] T. Wu, "SRP-6: Improvements and Refinements to the Secure Remote Password Protocol," *Submission to IEEE P1363 Working Group*, 2002.

[45] J. Zhou and K.-Y. Lam, "Undeniable billing in mobile communications," in *Proceedings of the 4th ACM/IEEE International Conference on Mobile Computing and Networking*, Dallas, Texas, October 1998.

[46] T. Aura, "Strategies against replay attacks," in *Proceedings of the 10th IEEE Computer Society Foundations Workshop*, pages 59-68, June 1997.

[47] F. Bao, "Analysis of a Conference Scheme Under Active and Passive Attacks," in *Proc. ACISP '04*, pp. 157-163, 2004.

[48] M.-S. Hwang and W.-P. Yang, "Conference key distribution schemes for secure digital mobile communications," *IEEE Journal of Selected Areas in Communications*, 13(2):416-420, February 1995.

[49] M.-S. Hwang, "Dynamic participation in a secure conference scheme for mobile communications," *IEEE Trans. Veh. Technol.*, 48(5):1469-1474, Sept. 1999.

[50] K. F. Hwang and C. C. Chang, "A self-encryption mechanism for authentication of roaming and teleconference services," *IEEE Trans. on Wireless Communications*, 2(2):400-407, March, 2003.

[51] I. Ingemarsson, D. T. Tang, and C. K. Wong, "A conference key distribution system," *IEEE Trans. Inform. Theory*, 28(5):714-720, Sept. 1982.

[52] W.-C. Ku, H.-C. Tsai, and S.-M. Chen, "Two simple attacks on Lin-Shen-Hwang's strong-password authentication protocol," *ACM SIGOPS Operating Systems Review*, 37(4):26-31, October 2003.

[53] S. Malladi, J. Alves-Foss, and R. B. Heckendorn, "On preventing replay attacks on security protocols," in *Proceedings of International Conference on Security and Management*,pages 77-83, June 2002.

[54] C. J. Mitchell and L. Chen, "Comments on the S/KEY user authentication scheme," *ACM SIGOPS Operating Systems Review*, v.30 n.4, p.12-16, October 1996.

[55] S. L. Ng, "Comments on 'Dynamic participation in a secure conference scheme for mobile communications'," *IEEE Trans. Veh. Technol.*, 50(1):334-335, Jan. 2001.

[56] C. Park, K. Kurosawa, T. Okamoto, and S. Tsujii, "On key distribution and authentication in mobile radio networks," in *Proceedings of Eurocrypt '93*, pp. 131-138, 1993.

[57] P. F. Syverson, "On key distribution protocols for repeated authentication," *Operating Systems Review*, 27(4):24-30, October 1993.

[58] P. F. Syverson, "A taxonomy of replay attacks," in *Proceedings of IEEE Computer Security Foundations Workshop VII*, pages 187-191, June 1994.

[59] M. Tatebayashi, N. Matsuzaki, and J. D. B. Newman, "Key distribution protocol for digital mobile communication systems," in *Proc. Crypto '89*, pp. 324-334, 1989.

[60] X. Yi, C. K. Siew, and C. H. Tan, "A secure and efficient conference scheme for mobile communications," *IEEE Transactions on Vehicular Communications*, 52(4):784-793, Jul. 2003.

[61] X. Yi, C. K. Siew, C. H. Tan, and Y. Ye, "A secure conference scheme for mobile communications," *IEEE Transactions on Wireless Communications*, 2(6):1168-1177, Nov. 2003.

[62] 3GPP TS 33.102, "3rd Generation Parnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security Architecture," December 2002.

[63] C. Perkins, E. M. Royer, and S. R. Das, "Ad hoc on-demand distance vector (AODV) routing," draft-ietf-manet-aodv-06.txt, July 2000.

[64] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad-hoc network routing protocols," in *Proceedings of the 4th ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom'98)*, October 1998.

[65] M. Gerla, K. Tang, and R. Bagrodia, "TCP performance in wireless multi-hop networks," in *Proceedings of IEEE WMCSA'99*, February 1999.

[66] G. Holland and N. H. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *Proceedings of ACM Mobicom'99*, pages 219-230, 1999.

[67] M. Woo, S. Singh, and C. S. Raghavendra, "Power-aware routing in mobile ad hoc networks," in *Proceedings of Mobicom'98*, October 1998.

[68] E. Çelebi, "Performance evaluation of wireless multi-hop ad-hoc network routing protocols," *Master thesis, Boğaziçi University*, 2002.

[69] D. McGrew and A. Sherman, "Key Establishment in Large Dynamic Groups Using One-way Function Trees," *Techinical Report* 0755, TIS Labs at Network Associates, Inc., May 1998.

[70] C. K. Wong, M. Gouda and S. Lam, "Secure Group Communications Using Key Graphs," In *Proceedings of SIGCOMM*, 1998.

[71] C. K. Wong, M. Gouda, and S. Lam, "Secure Group Communications Using Key Graphs," *IEEE/ACM Transactions on Networking*, 8(1):16-30, Feb. 2000.

[72] D. Balenson, D. McGrew, and A. Sherman, "Key Mangement for Large Dynamic Groups: One-way Function Trees and Amortized Initialization," *Internet draft, IETF*, June 2002.

[73] E. Harder, D. M. Wallner, and R. C. Agee, "Key Management for Multicast: Issues and Architectures," *IETF RFC 2627*, 1999.

[74] A. Perrig, D. Song, and D. Tygar, "ELK, a New Protocol for Efficient Large-Group Key Distribution," in*Proceedings of IEEE Security and Privacy Symposium 2001*, May 2001.

[75] Y. Kim, A. Perrig, and G. Tsudik, "Simple and fault-tolerant key agreement for dynamic collaborative groups," in *Proceedings of the 7th ACM Conference on Computer and Communications Security*, pages 235-244, November 2000.

[76] Y. Kim, A. Perrig, and G. Tsudik, "Communication-efficient group key agreement," in *Proceedings of IFIP SEC 2001*, June 2001.

[77] Y. Kim, A. Perrig, and G. Tsudik, "Tree based group key agreement," *ACM Transactions on Information Systems Security*, to appear in 2004.

[78] W.-H. Yang and S.-P. Shieh, "Secure key agreement for group communications," *ACM/PH international journal of network management*, 11(6):365-374, Nov.-Dec., 2001.

[79] M. Steiner, G. Tsudik, and M. Waidner, "Key agreement in dynamic peer groups," *IEEE Transactions on Parallel and Distributed Systems*, August 2000.

[80] H. Harney and E. Harder, "Logical Key Hierarchy Protocol," *Internet draft, IETF*, April 1999.

[81] Y. Amir, Y. Kim, C. Nita-Rotaru, and G. Tsudik, "On the performance of group key agreement protocols," in *Proceedings of ICDCS'02*, pages 463-464, July 2002.

[82] K. Chen and K. Nahrstedt, "Effective location-guided tree construction algorithms for small group multicast in MANET," in *Proceedings of IEEE Infocom'02*, 2002.

[83] C. Gui and P. Mohapatra, "Efficient overlay multicast for mobile ad hoc networks," in *Proceedings of IEEE WCNC 2003*, March 2003.

[84] C. Gui and P. Mohapatra, "Scalable multicasting in mobile ad hoc networks," in *Proceedings of IEEE INFOCOM'04*, March 2004.

[85] Y. Zhu and T. Kunz, "MAODV Implementation for NS-2.26," Technical Report SCE-04-01, Carleton University, January 2004.

[86] M. Scott, "Multiprecision Integer and Rational Arithmetic C/C++ Library," Shamus Software Ltd, available: http://indigo.ie/∼ mscott/

[87] L. Lazos, J. Salido, and R. Poovendran, "VP3: Using Vertex Path and Power Proximity for Energy Efficient Key Distribution," in *Proceedings of VTC'04*, 2004.

[88] L. Lazos and R. Poovendran, "Energy-Aware Secure Multicast Communication in Wireless Ad-Hoc Networks," *Technical Report UWEETR-2003-0013*, University of Washington, 2003.

[89] L. Lazos and R. Poovendran, "Cross-Layer Design for Energy-Efficient Secure Multicast Communications in Ad Hoc Networks," in *Proceedings of ICC'04*, 2004.

[90] Y. Sun, W. Trappe, and K.J. Liu, "A Scalable Multicast Key Management Scheme for Heterogeneous Wireless Networks," *IEEE/ACM Transactions onn Networking*, August 2004.

[91] J. Chuang and M. Sirbu, "Pricing Multicast Communications: A Cost-Based Approach," *Telecommunication Systems* 17 (3):281-297, July 2001.

[92] M. Abdalla, P. Fouque and D. Pointcheval, "Password-Based Authenticated Key Exchange in the Three-Party Setting," *IACR eprint 2004/233*, available: http://eprint.iacr.org/2004/233/

[93] N. Asokan and P. Ginzboorg, "Key Agreement in Ad-hoc Networks," *Computer Communications*, 23(18):1627-1637, 2000.

[94] F. Bao, "Security Analysis of a Password Authenticated Key Exchange Protocol," in *Proceedings of ISC 2003*, 2003.

[95] J. W. Byun and D. H. Lee, "N-Party Encrypted Diffie-Hellman Key Exchange Using Different Passwords," *Proceedings of ACNS 2005*, LNCS 3531, pages 75-90, 2005.

[96] M. Bellare and P. Rogaway, "The AuthA Protocol for Password-Based Authenticated Key Exchange," *Contribution to the IEEE P1363 study group*, March 2000.

[97] M. Bellare, D. Pointcheval and P. Rogaway, "Authenticated Key Exchange Secure Against Dictionary Attack," *Advances in Cryptology - EUROCRYPT 2000*, Lecture Notes in Computer Science, vol. 1807, pp. 139-155, Springer-Verlag, May 2000.

[98] S. M. Bellovin and M. Merritt, "Encrypted Key Exchange: Password Based Protocols Secure against Dictionary Attacks," In *Proceedings 1992 IEEE Symposium on Research in Security and Privacy*, pages 72-84. IEEE Computer Society, 1992.

[99] S. M. Bellovin and M. Merritt, "Augmented EncryptedKey Exchange: A Password-based Protocol Secure against Dictionary attacks and Password File Compromise," in *Proceedings of the 1st ACM Conference on Computer and Communication Security*, pages 244-250, 1993.

[100] V. Boyko, P. D. MacKenzie, S. Patel, "Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman," *EUROCRYPT 2000*: 156-171.

[101] E. Bresson, O. Chevassut and D. Pointcheval, "Group Diffie-Hellman Key Exchange Secure against Dictionary Attacks," in *Proceedings of Asiacrypt 2002*, December 2002.

[102] M. Burmester and Y. Desmedt, "A Secure and Efficient Conference Key Distribution System (extended abstract)," in *Advances in Cryptology-Eurocrypt 94*, Lecture Notes in Computer Science, vol. 950. Springer-Verlag, New York, 1994.

[103] J. W. Byun, I. R. Jeong, D. H. Lee, and C.-S. Park, "Password-Authenticated Key Exchange between Clients with Different Passwords," in *Proceedings of ICICS 2002*, pp. 134-146, 2002.

[104] R. Gennaro, Y. Lindell, "A Framework for Password-Based Authenticated Key Exchange," *EUROCRYPT 2003*: 524-543, available: http://eprint.iacr.org/2003/032/

[105] Oded Goldreich, Yehuda Lindell, "Session-Key Generation Using Human Passwords Only," *CRYPTO 2001*: 408-432, available: http://eprint.iacr.org/2000/057/

[106] D. P. Jablon, "Extended Password Key Exchange Protocols Immune to Dictionary Attacks," *WETICE 1997*: 248-255, IEEE Computer Society, June 18-20, 1997.

[107] J. Katz, R. Ostrovsky, and M. Yung, "Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords," *EUROCRYPT 2001*: 475-494, 2001.

[108] J. Katz, R. Ostrovsky, and M. Yung, "Forward Security in Password-Only Key Exchange Protocols," in *Proceedings of Security in Communication Networks 2002 conference (SCN'02)*, Springer-Verlag Lecture Notes in Computer Science, 2002.

[109] T. Kwon, "Summary of AMP, Contribution for the P1363 standard," available: http://grouper.ieee.org/groups/1363/passwdPK/contributions/ampsummary.pdf, August 2003.

[110] T. Kwon, "Addendum to Summary of AMP, Contribution for the P1363 standard," available: http://grouper.ieee.org/groups/1363/passwdPK/contributions/ampsummary2.pdf, November 2003.

[111] C.-L. Lin, Hung-Min Sun, and T. Hwang, "Three-party Encrypted Key Exchange: Attacks and A Solution," *ACM Operating Systems Review*, 34(4):12-20, 2000.

[112] C.-L. Lin, H.-M. Sun, and T. Hwang, "Three-party Encrypted Key Exchange Without Server Public-Keys," *IEEE Communications Letters*, 5(12):497-499, December 2001.

[113] S. Lucks, "Open Key Exchange: How to Defeat Dictionary Attacks Without Encrypting Public Keys," *Security Protocols Workshop* 1997: 79-90.

[114] P. MacKenzie, "The PAK suite: Protocols for Password-Authenticated Key Exchange," *Submission to IEEE P1363.2*, April 2002.

[115] P. MacKenzie, S. Patel, and R. Swaminathan, "Password-Authenticated Key Exchange Based on RSA," in *Proceedings of AsiaCrypt 2000*, pages 599-613, LNCS, Springer-Verlag, 2000.

[116] P. MacKenzie, "The PAK Suite: Protocols for Password-Authenticated Key Exchange," *Submission to IEEE P1363.2*, April 2002.

[117] D. Steer, L. Strawczynski,W. Diffie, and M. Wiener, "A Secure Audio Teleconference System," in S. Goldwasser, editor, *Advances in Cryptology - CRYPTO88*, 1988.

[118] M. Steiner, G. Tsudik and M. Waidner, "Refinement and Extension of Encrypted Key Exchange," *ACM SIGOPS Operating Systems Review*, 29(3):22-30, 1995.

[119] M. Steiner, G. Tsudik and M. Waidner, "Diffie-Hellman Key Distribution Extended to Group Communication," in *Proceedings of the 3rd ACM Conference on Computer and Communication Security*, March 1996.

[120] M. Steiner, G. Tsudik and M. Waidner,"Cliques: A New Approach to Group Key Agreement," *IEEE TPDS*, August 2000.

[121] M. Steiner, G. Tsudik and M. Waidner, "Key Agreement in Dynamic Peer Groups," *IEEE Transactions on Parallel and Distributed Systems*, August 2000.

[122] T. Wu. "The Secure Remote Password Protocol," in *1998 Internet Society Symposium on Network and Distributed System Security*, pages 97-111, 1998.

[123] F. Zhu, D. S. Wong, A. H. Chan, and R. Ye, "Password authenticated key exchange based on RSA for imbalanced wireless networks," in *Proceedings of ISC 2002*, LNCS 2433, papes 150-161, Springer-Verlag, 2002.

[124] J. Kong, X. Hong, M.Y. Sanadidi, M. Gerla, "Mobility Changes Anonymity: Mobile Ad Hoc Networks Need Efficient Anonymous Routing", The Tenth IEEE Symposium on Computers and Communications (ISCC), June 27-30, 2005.

[125] A. Pfitzmann and M. Köhntopp, "Anonymity, Unobservability, and Pseudonymity: A Consolidated Proposal for Terminology", available: http://dud.inf.tu-dresden.de/literatur /Anon_Terminology_v0.22.pdf. Draft, version 0.22, July 2000.